Apertus: Democratizing Open and Compliant LLMs for Global Language Environments

Alejandro Hernández-Cano[1], Alexander Hägele[1], Allen Hao Huang[1], Angelika Romanou[1]
Antoni-Joan Solergibert[1,2], Barna Pásztor[2], Bettina Messmer[1], Dhia Garbaya[1]
Eduard Frank Ďurech[1,2], Ido Hakimi[2], Juan Garcia Giraldo[1], Mete Ismayilzada[1]
Negar Foroutan[1], Skander Moalla[1], Tiancheng Chen[2], Vinko Sabolčec[1]
Yixuan Xu[1,2], Michael Aerni[2], Badr AlKhamissi[1], Inés Altemir Marinas[1]
Mohammad Hossein Amani[1], Matin Ansaripour[1], Ilia Badanin[1,2], Harold Benoit[1]
Emanuela Boros[1], Nicholas John Browning[3], Fabian Bösch[3], Maximilian Böther[2]
Niklas Canova[2], Camille Challier[1], Clément Charmillot[1], Jonathan Coles[3]
Jan Milan Deriu[7], Arnout Devos[2], Lukas Drescher[3], Daniil Dzenhaliou[1]
Maud Ehrmann[1], Dongyang Fan[1], Simin Fan[1], Silin Gao[1], Miguel Gila[3]
María Grandury[1], Diba Hashemi[1], Alexander Miserlis Hoyle[2], Jiaming Jiang[1]
Mark Klein[3], Andrei Kucharavy[4], Anastasiia Kucherenko[4], Frederike Lübeck[2]
Roman Machacek[9], Theofilos Ioannis Manitaras[3], Andreas Marfurt[5], Kyle Matoba[1]
Simon Matrenok[1], Henrique Mendonça[3], Fawzi Roberto Mohamed[3], Syrielle Montariol[1]
Luca Mouchel[1], Sven Najem-Meyer[1], Jingwei Ni[2], Gennaro Oliva[3], Matteo Pagliardini[1]
Elia Palme[3], Andrei Panferov[6], Léo Paoletti[1], Marco Passerini[3], Ivan Pavlov[1]
Auguste Poiroux[1], Kaustubh Ponkshe[1], Nathan Ranchin[1], Javier Rando[2], Mathieu Sauser[1]
Jakhongir Saydaliev[1], Mukhammadali Sayfiddinov[2], Marian Schneider[2]
Stefano Schuppli[3], Marco Scialanga[1], Andrei Semenov[1], Kumar Shridhar[2]
Raghav Singhal[1], Anna Sotnikova[1], Alexander Sternfeld[4], Ayush Kumar Tarun[1]
Paul Teiletche[1], Jannis Vamvas[8], Xiaozhe Yao[2], Hao Zhao[1], Alexander Ilic[2]
Ana Klimovic[2], Andreas Krause[2], Caglar Gulcehre[1], David Rosenthal[10], Elliott Ash[2]
Florian Tramèr[2], Joost VandeVondele[3], Livio Veraldi[10], Martin Rajman[1]
Thomas C. Schulthess[3], Torsten Hoefler[2], Antoine Bosselut[1], Martin Jaggi[1], Imanol Schlag[2]

[1]EPFL, [2]ETH Zurich, [3]CSCS, [4]HES-SO Valais-Wallis, [5]HSLU, [6]IST Austria
[7]ZHAW, [8]University of Zurich, [9]University of Bern, [10]Vischer

Abstract

Open LLMs enable AI practitioners to control development costs by building on an existing foundation for downstream applications. While offering substantial promise, current models often fail to meet the needs of users needing open solutions aligned with responsible AI principles, including data compliance, transparency, and inclusivity. In this work, we present Apertus, a fully open suite of large language models (LLMs) designed to address responsibility shortcomings in today's open model ecosystem, namely, data responsibility and global representation. Unlike many prior models that release weights without reproducible data pipelines or regard for content-owner rights, Apertus models are pretrained exclusively on openly available data, retroactively respecting robots.txt exclusions and filtering for non-permissive, toxic, and personally identifiable content. To mitigate risks of data memorization, we also adopt the Goldfish objective during pretraining, strongly suppressing verbatim recall of data while retaining downstream task performance. Apertus also drastically expands multilingual coverage, training on 15T tokens from over 1800 languages, with ~40% of pretraining data allocated to non-English content. Released at 8B and 70B scales, Apertus approaches state-of-the-art results among fully open models on multilingual benchmarks, rivalling or surpassing open-weight counterparts.[1]

1 Introduction

An expansive open ecosystem for developing large language models (LLMs) has flourished since the

[1] http://apertus-ai.org/

release of GPT-J (Wang and Komatsuzaki, 2021), with the quality of released models improving and accelerating (Black et al., 2022; Zhang et al., 2022; Scao et al., 2022; Touvron et al., 2023a,b; Jiang et al., 2023; Bai et al., 2023; Mesnard et al., 2024; Grattafiori et al., 2024; Yang et al., 2024a; Riviere et al., 2024; Yang et al., 2024b; Kamath et al., 2025; Yang et al., 2025). Despite this proliferation of new, powerful LLMs, many of them continue to overlook the needs of many global users that require responsible AI foundations. At various points throughout the LLM development pipeline, design decisions introduce systemic limitations that hinder further downstream development for many users.

We release the Apertus suite of models to address several of these limitations — in particular, responsible data practices and multilingual representation — to help democratize LLMs for broader communities of global users. First, we set new standards for data compliance. Most of today's open models are, in fact, not open-source or reproducible, but only open-weight (Jiang et al., 2023; Grattafiori et al., 2024; Kamath et al., 2025, *inter alia*), with offerings by a few organizations (*e.g.*, EleutherAI, Allen AI, LLM360, BigScience, etc.) serving as notable exceptions (Black et al., 2022; Scao et al., 2022; Liu et al., 2024b; Groeneveld et al., 2024, *inter alia*). Open-weight models do not release the data used for training the model and often reveal very little about it beyond the token count. Simultaneously, many of these open-weight models allegedly include large amounts of illegal material that do not consider the access rights granted by content owners.[2] By contrast, we pretrain Apertus solely on openly available data sources, with documents excluded whenever their owners have opted out of AI crawling through `robots.txt` (Fan et al., 2025). We also train Apertus using a variant of the Goldfish objective (Hans et al., 2024) to limit the memorization and possible reproduction of training data. Our evaluation, the first at this scale, demonstrates that this approach effectively prevents verbatim memorization of training data while preserving downstream task performance.

Second, we focus on expanding the multilingual representation of Apertus. Most models today only focus on single languages (Touvron et al., 2023b; Mesnard et al., 2024; Liu et al., 2025a), or small subsets of high-resource languages (Yang et al., 2024b; Grattafiori et al., 2024; Kamath et al., 2025), limiting their extensions for lower-resource language environments.[3] For Apertus, we massively expand the number of languages represented in our pretraining data, to over 1800 languages, and set aside a much larger proportion of our pretraining text data mixture (∼40%) for non-English languages. We also include over 149 languages in our post-training mixture.

This work describes our Apertus models, a collection of pretrained and, in line with prior work (Lambert et al., 2025; Martins et al., 2025), Instruct models at 8B and 70B scale. To train the first fully open 70B-parameter model at this scale, we implemented several architectural innovations (*e.g.*, xIELU) and training advances (*e.g.*, AdEMAMix, QRPO) to stabilize large-scale training. The models were pretrained on 15T tokens using up to 4096 GPUs. While our main body summarize our core contributions in data compliance, multilinguality, memory prevention, and architectures, comprehensive details of our design decisions, ablations studies, and training procedure, including architecture (Appx. C), pretraining data (Appx. D), post-training (Appx. E), and infrastructure (Appx. G), are provided in the Appendix as a valuable resource to the community for future development.

Our analysis (Section 5) evaluate the impact of these responsible design choice, and show that the Apertus models are the strongest pretrained open models on multilingual benchmarks with open state-of-the-art performance at equivalent scale, even outperforming solely open-weight counterparts in several settings. We summarize our unique contributions as the following:

- **Responsible Data Practices.** The pretraining corpus was compiled solely from public web data, respecting `robots.txt` not only at crawl time (January 2025), but also retroactively applying January 2025 opt-out preferences to web scrapes from previous crawls. All datasets used for post-training were similarly filtered for non-compliant data (*e.g.*, data with non-permissive licenses). These filters are designed to comply with data provisions of the EU AI Act and similar regulations.

- **Memorization Prevention.** The Apertus

[2] www.theatlantic.com/technology/archive/2025/03/libgen-meta-openai/682093

[3] The BLOOM (Scao et al., 2022), Aya (Üstün et al., 2024), and Qwen3 (Yang et al., 2025) models are exemplary exceptions to this practice. They train on more languages, but still ∼10× fewer than in our work.

models are pretrained using the Goldfish objective (Hans et al., 2024), limiting the model's ability to regurgitate training data.

- **Multilinguality.** We train our model on 15T tokens from 1811 identified languages during pretraining, taken from the FineWeb-2 web crawl dataset.[4] We use data from 149 languages in post-training. We test our models on cultural, knowledge, and instruction-following benchmarks covering 94 languages.

We will release all scientific artifacts from our development cycle with a permissive license, including data preparation scripts, checkpoints, evaluation suites, and training code, enabling transparent audit and extension. This commitment to transparency grounds our model's name "*Apertus*", Latin for "*open*". Apertus is the leading fully open LLM today, appropriate for a broad range of use cases, manifesting the first release of our vision of world-class responsible LLMs for global use.

2 Responsible Data Construction

This section covers responsible data considerations for our pretraining and post-training.

2.1 Pretraining Data

Consent: `robots.txt` with Hindsight. Pretraining datasets based on web data are typically constructed by aggregating multiple samples of web crawls at different points in time (Penedo et al., 2024a, 2025). To prevent their content from being crawled, content owners may apply restrictions on web crawlers by updating their `robots.txt` files (Longpre et al., 2024b; Fan et al., 2025). However, pretraining datasets, when they account for these restrictions at all, typically enforce them only at the moment of crawling. This practice raises concerns about data usage, as subsequent changes to access policies are not retroactively applied to previously collected web snapshots, potentially leading to the continued use of data that is no longer permitted under the updated restrictions. To respect the consent of data owners, we retroactively apply the most recent crawling permissions specified by data owners. This filter is applied to *all* datasets.

To implement this filter, we begin by ranking URL domains according to the volume of texts they contribute to the FineWeb (Penedo et al., 2024a) and FineWeb-2 (Penedo et al., 2025) corpus, as an approximation of web-level English and multilingual data. From this ranking, we select the top one million English domains and the top one million non-English domains. For each domain that remains reachable (some sites are now offline), we retrieve its `robots.txt` file as of January 2025 and examine the directives relevant to AI training. In particular, we focus on those targeting the AI-specific user agents listed in Appendix D.5. Any contents blocked by the current `robots.txt` is removed retroactively from the entire 2013-2024 range of the training dataset. We follow an opt-out policy, that is, if the corresponding `robots.txt` files are not available, we consider the data usable for training. The filtering process results in an estimated token loss of approximately 8% in English data and 4% in multilingual data.

Personally Identifiable Information (PII). To protect against potential memorization of PII in the model, we anonymize pretraining data using best-effort practices to process data on the scale of hundreds of terabytes of data (Penedo et al., 2024a, 2025). We apply regular expressions to detect email addresses, IP addresses, and IBAN bank account numbers, and replace them with anonymous markers, such as `<email-pii>`.

Toxicity Filtering. We implement multilingual toxicity filtering across nine languages (English, Chinese, French, German, Italian, Dutch, Polish, Spanish, and Portuguese) on FineWeb-2 (Penedo et al., 2025) and FineWeb (Penedo et al., 2024a). To identify toxic content, we train language-specific binary classifiers using annotated datasets (PleIAs; Arnett et al., 2024, and SWSR; Jiang et al., 2021) and annotate toxicity scores for FineWeb-2 and FineWeb documents. We filter the 5% of documents per language with the highest predicted toxicity scores from the corpus.

2.2 Post-training Data

The collection and preparation of our post-training data follow the same core principles as our pretraining corpus: transparency, permissive licensing, multilingual inclusivity, and legal compliance. We begin by collecting openly available datasets, which we subject to legal and quality filtering. Selected datasets are then decontaminated against our evaluation benchmarks to ensure the integrity and reliability of downstream assessments.

[4] https://github.com/huggingface/fineweb-2/blob/main/fineweb2-language-distribution.csv

License Filtering. For post-training, we collect a broad set of candidate datasets and filter them according two responsibility criteria: (i) content must be explicitly released under licenses permitting redistribution and commercial use (*e.g.*, CC-BY, Apache 2.0), ensuring republishable datasets, and (ii) the collection procedure must be fully documented and reproducible, ensuring versioned datasets. When performing license filtering, we distinguish between *source datasets* and *compound datasets* (or *mixtures*), which incorporate multiple source datasets or other mixtures. For source datasets, we manually filter datasets released under non-permissive or restrictive licenses (*e.g.*, NC), or those with ambiguous or unspecified licenses. For compound datasets, we verify that the overarching license of a mixture aligns with the licenses of all constituent source datasets and mixtures. In the rare cases where we detect invalid re-licensing, we exclude the dataset from our mixture. Likewise, we systematically exclude source datasets originating from providers that have opted out of AI training through `robots.txt`, possess share-alike licenses (*e.g.*, Reddit, StackExchange), or otherwise fail to meet our compliance standards. The impact of license filtering is evaluated along with decontamination (see Table 3 below).

Quality Filtering. We quality filter the datasets through a combination of metadata analysis and manual inspection. We rely on dataset metadata such as the provider, the scientific impact of the release, and, most importantly, whether the data is of human or synthetic origin as initial proxies of quality. We also manually inspect datasets for hallucinations in synthetic data, overly long or incoherent responses, and the presence of repetitive patterns in model outputs, filtering datasets that exhibit high degrees of these patterns. For math- and code-related tasks, we prioritise datasets with verified solutions. Manually selected datasets make it into our candidate mixtures, the best of which is determined by fine-tuning and evaluation iterations.

Decontamination. We decontaminate all datasets against the benchmarks used for evaluation. Following allal et al. (2025); Lambert et al. (2025); Walsh et al. (2025), we use n-gram matching to identify and remove training samples that are identical or similar to benchmark prompts (more details in Appendix E.1.2).

2.3 Multilinguality.

Across all data sources, we prioritize multilingual representation. For pretraining, we used FineWeb-2 and FineWeb-2-HQ as sources. For post-training, we used SmolTalk2 conversational data (1.3M examples across 8 languages), EuroBlocks synthetic multilingual instructions (157K), and language-specific datasets for post-training. Running Language ID on these datasets reveals that 1811 languages are represented in the pretraining data and 149 are in the post-training suite. More details can be found in Appendices D.1.2 and E.1.

3 Method

3.1 Model Architecture

Apertus is a dense decoder-only Transformer (Vaswani et al., 2017; Radford et al., 2018). The basic architecture consists of a deep stack of Transformer blocks, containing a multi-head self-attention mechanism and a feed-forward network (MLP), with residual connections and normalization applied around each sublayer. We initialize this architecture at two scales (1) **Apertus-8B**, with 32 layers and 32 parallel attention heads, and (2) **Apertus-70B**, with 80 layers and 64 parallel attention heads. The main attributes of the models are summarized in Table 5. We use established modifications to the original Transformer (*e.g.*, RoPE, RMSNorm), and improve architectural efficiency through the use of QK-Norms (Henry et al., 2020; Dehghani et al., 2023) and the xIELU activation function (Huang and Schlag, 2025). The following list describes each modification in more detail:

No biases. We remove all bias terms from the architecture (Chowdhery et al., 2023).

Pre-Norm and RMSNorm. We use pre-normalization before the residual in the transformer block.We replace LayerNorm (Ba et al., 2016) with RMSNorm (Zhang and Sennrich, 2019), which has equivalent performance while improving efficiency.

Rotary Positional Embeddings. We use RoPE embeddings (Su et al., 2024) with a base $\Theta = 500,000$ during pretraining, which we extend in the long-context phase (Section C.6). We also employ NTK-aware RoPE scaling (Peng et al., 2024).

Group-Query Attention. For inference efficiency, we adopt grouped-query attention (GQA; Ainslie et al., 2023), which uses fewer key-value pairs than query heads without compromising performance.

Untied Embeddings and Output Weights. Input embeddings are not tied to output weights, improv-

Model	Layers	Dim	MLP Dim	Heads (Q / KV)	Activation	Context Max Length	Sequence Length	Batch Size (Tokens)	Training Steps	Peak LR	Training Tokens
Apertus 8B	32	4096	21504	32/8	xIELU	65536	4096	4.2M → 8.4M	2.6M	1.1e-4	15T
Apertus 70B	80	8192	43008	64/8	xIELU	65536	4096	8.4M → 16.8M	1.1M	1.0e-5	15T

Table 1: **Apertus Model Architecture and Training Hyperparameters Overview.** We train our custom Apertus architecture across two scales, 8B and 70B. For both models, we double the global batch size in middle stages of training. More detailed hyperparameters are provided in Table 11.

ing performance at the cost of additional memory.

QK-Norm. We incorporate QK-Norm (Henry et al., 2020; Dehghani et al., 2023), which normalizes the queries and keys in the attention layers. QK-Norm improves training stability by preventing excessively large attention logits.

xIELU. In MLP sublayers, we adopt the xIELU activation function (Huang and Schlag, 2025).

Context length. Both Apertus models were trained with a context of 4096 tokens during pretraining. We then perform a long-context extension to support sequences of up to 65536 tokens.

3.2 Tokenizer

We adapt the `Mistral-Nemo-Base-2407` tokenizer, which accommodates multilingual text and code.[5] The vocabulary has 2^{17}=131,072 tokens, of which we modified 47 special tokens to better support code and math data. We based our choice on a comparison of several tokenizers (*e.g.*, Llama-3.1, Mistral-Nemo, Qwen-2.5, and Gemma-2) using four intrinsic metrics, which showed that the Mistral-Nemo tokenizer matches or exceeds comparable tokenizers on all metrics (§C.2).

3.3 Pretraining

We introduce multiple changes to current pretraining recipes to prevent memorization (using the Goldfish loss; Hans et al., 2024), improve training efficiency (with AdEMAMix; Pagliardini et al., 2025), and facilitate continual training (WSD learning schedule; Zhai et al., 2022; Hu et al., 2024).

Training Objective. Verbatim regurgitation of training data is a significant concern in LLMs, raising both copyright (Chang et al., 2023; Karamolegkou et al., 2023) and privacy risks (Huang et al., 2022). Consequently, we adopt the goldfish loss, which reduces memorization while having minimal impact on performance (Hans et al., 2024). Specifically, the goldfish loss computes the language modeling objective on only a subset of tokens based on a mask $G \in \{0,1\}^L$:

$$\mathcal{L}(\boldsymbol{\theta}) = -\frac{1}{|G|}\sum_{t=1}^{L} G_t(x_t)\log P_{\boldsymbol{\theta}}(x_t|x_{<t}),$$

where L is the sequence length, x_t is the t-th token and $x_{<t}$ is the preceding context. The binary mask G is randomly sampled for each batch during training.[6] In practice, we front-load token masking during data loading rather than during pretraining for efficiency. Following Xu (2025), we use a 2% token masking rate, which effectively suppresses verbatim memorization without compromising downstream performance.[7]

AdEMAMix. We pretrain using the AdEMAMix optimizer (Pagliardini et al., 2025), which is a first for an LLM at this scale. AdEMAMix improves upon existing optimizers that rely on Exponential Moving Averages (EMA) of gradients (*e.g.*, Adam; Kingma and Ba, 2015; Loshchilov and Hutter, 2017) by adding a long-term EMA as an additional momentum vector. Recent optimizer benchmarking show AdEMAMix scales more favourably with model size, training duration, and batch size than other optimizers (Semenov et al., 2025).

Learning Rate Schedule. We employ the Warmup-Stable-Decay (WSD) learning rate (LR) schedule (Hu et al., 2024; Zhai et al., 2022), which enables downstream continual pretraining without rewarming the learning rate since the full training duration does not have to be specified in advance (Hägele et al., 2024; Schaipp et al., 2025).[8] Our LR warmup starts from 10% of the peak LR and is linearly increased for 16.8B tokens.

For the LR decay stage, we use a negative square root shape, which reliably outperforms a standard linear shape (Hägele et al., 2024; Dremov et al., 2025). For both model sizes, the decay period coincides with a change in the data mixture for the

[5] https://huggingface.co/mistralai/Mistral-Nemo-Base-2407

[6] Algorithm 1 in the Appendix details our implementation.

[7] Ablations in Appendix Figure 4 and Table 7.

[8] In fact, during pretraining, we extended the initial planned training phase from 9T to 15T tokens with no schedule change.

highest-quality sources at 13.5T consumed tokens (§ D). The final learning rate is set to a factor of 10% of the respective peak LR to facilitate downstream stable finetuning (*i.e.*, long context extension and post-training).

Batch Size and Sequence Length. To maximise efficiency, we employ a sequence length of 4096 tokens and an initial batch size of 1024 (4.2M tokens) and 2048 (8.4M tokens) for the 8B and 70B models, respectively. After 8T tokens for the 8B model and 4.4T for the 70B, we intentionally doubled both the number of nodes and the batch size at this stage, while keeping the learning rate unchanged.[9]

3.4 Post-training

Long Context. In the first stage of post-training, we train our models to handle extended context lengths. We split training into multiple phases to adapt the maximum context length iteratively and smoothly, avoiding the performance instability that can result from a sudden, drastic increase in context length. Further detail around these phases (§C.6), the data mixture during long context extension (§D.4), and our long-context evaluations (§F.2) can be found in the Appendix.

SFT. We then perform a supervised finetuning phase using a mixture outlined in Appendix E.1. For Apertus-8B and Apertus-70B, we use a global batch size of 512 and 1024, and learning rates of 5×10^{-6} and 2×10^{-6}, respectively, with a linear decay schedule. All models are trained with a maximum sequence length of 4096 tokens, and the AdEMAMix optimizer (Pagliardini et al., 2025) with $\beta_3 = 0.99$ (different from pretraining), $\alpha = 8.0$, and both t_{β_3} and t_α set to the total number of training steps. The values $\beta_1 = 0.9$ and $\beta_2 = 0.999$ match the ones used in pretraining.

Preference Alignment. After SFT, our alignment pipeline shapes the model's behavior according to helpfulness, honesty, safety, and refusal. We adopt the recently-proposed Quantile Reward Policy Optimization algorithm (QRPO; Matrenok et al., 2025), which optimizes an absolute reward signal by minimizing the following loss:

$$\mathcal{L}_{QRPO} = \mathop{\mathbb{E}}_{x,y}\Big[\big(\mathcal{R}_q(x,y) - \beta_{KL}\log Z_q(x) - \beta_{KL}\log\tfrac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}\big)^2\Big],$$

where $\mathcal{R}_q(x,y)$ is the quantile reward, representing the percentile rank of a candidate completion y among a set of reference completions (sampled from a reference policy π_{ref}), and $Z_q(x)$ is the corresponding partition function.[10]

We train the model using a dataset $\mathcal{D} = (x_i, y_i)$ composed of both offline completions (generated by other LLMs) and off-policy completions (generated by the reference model π_{ref}). For each prompt x_i, we generate a set of $n = 10$ reference completions $y_{i,j} \sim \pi_{ref}(\cdot \mid x_i)$, which are used both for training and to estimate the quantile reward. Each reference completion is annotated with a reward to construct the reference reward set:

$$\mathcal{S}_{ref,i} = \{\mathcal{R}(x_i, y_{i,j})\}_{j=1}^{n}.$$

The quantile reward $\mathcal{R}_q(x_i, y_i)$ is then computed as the empirical cumulative distribution function (CDF) of the reward over this reference set:

$$\mathcal{R}_q(x_i, y_i) = \frac{1}{|\mathcal{S}_{ref,i}|}\sum_{\mathcal{R}(x_i,y_{i,j})\in\mathcal{S}_{ref,i}} \mathbf{1}\left\{\mathcal{R}(x_i, y_{i,j}) \le \mathcal{R}(x_i, y_i)\right\}.$$

Rewards are provided either by a pretrained reward model or via an LLM-as-a-judge, assigning absolute scores for single completions or pairwise rankings (§E.4.1, §E.4.2).

4 Results

We evaluate Apertus models on a suite of multilingual and English benchmarks both during pretraining and after post-training. Following standard practice, we use the `lm-evaluation-harness` framework with the likelihood-scoring mode for pretraining evaluation, and the open-generation mode for post-training evaluation.

Pretraining Results. During pretraining, we continuously evaluate model checkpoints to monitor model improvements on a diverse collection of benchmarks (outlined in §F.1). Figure 1 shows the evolution of macro-averaged accuracy throughout training. Apertus models exhibit steady improvements across all evaluation sets, achieving strong multilingual performance (Global), while maintaining competitive results on English benchmarks.

[9] This results in minimal throughput degradation, as shown in Figure 13. Increasing the batch size has also been shown to be beneficial in later stages of training by increasing hardware efficiency, allowing training models that perform better under the same FLOP budget (Smith et al., 2018; McCandlish et al., 2018; Merrill et al.).

[10] We expound on these terms in Appendix E.4

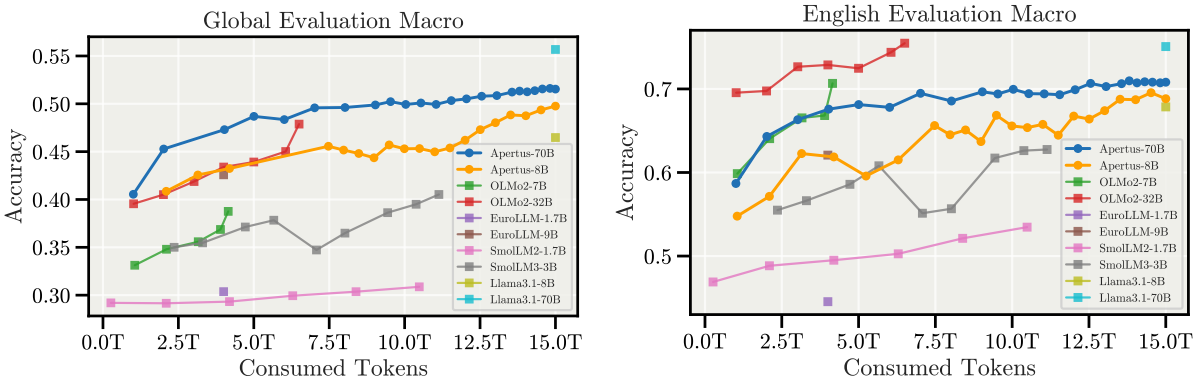

Figure 1: **Pretraining Evaluation Throughout Training.** Comparison of downstream evaluation results across model checkpoints as training progresses. Global Evaluation uses the full suite of evaluation benchmarks. English includes only English-language tasks. For Global, the aggregation between different benchmarks consists of a macro aggregation, where each different language of each dataset is considered as a separate datapoint to aggregate.

Post-training Results. We assess the post-trained Apertus models across a suite of benchmarks that capture key aspects of knowledge recall, reasoning, mathematics, coding, instruction following, and cultural knowledge. Table 2 summarizes the results, reporting category-wise averages over all benchmarks. Individual results for each benchmark are shown in Tables 27-31. Both Apertus-8B-Instruct and Apertus-70B-Instruct demonstrate strong overall performance, particularly in comparison to other fully open models of similar sizes. Notably, Apertus-8B-Instruct is the strongest model among fully open models of similar size in knowledge recall (Tab. 2, 27), commonsense reasoning (Tab. 2, 27), instruction following (Tab. 2, 29), and cultural knowledge (Tab. 2, 30). In cultural knowledge (Tab. 30), it is not only leading among fully open models but also approaches or surpasses the strongest models in its size class, such as Qwen3-8B. Performance in math and coding is comparatively weaker for both Apertus models (Tab. 2, 28), though most other baselines have undergone additional RL training (*e.g.*, RLVR), which is known to enhance these capabilities but has not yet been applied to Apertus. Detailed benchmark results by language are shown in Tab. 33-41, though we note that it can be difficult to draw comparisons between languages unless the benchmark has parallel data across languages.

5 Analysis

This section focuses on selected results regarding our core contributions: data compliance & memorization prevention. We evaluate the impact of data compliance in English and multilingual post-training, as well as memorization prevention on pretrained models. More ablations and evaluations can be found in our extensive Appendix.

Data Compliance. To quantify the impact of our data filtering approaches in post-training, we conducted an ablation study using Apertus-8B initialized from a 10T token checkpoint and finetuned on different data configurations. Table 3 presents results across 13 benchmarks, comparing four configurations: (1) original Tulu3 without filtering,[11] (2) Tulu3 with decontamination only, (3) Tulu3 with both decontamination and license filtering.

Our results show that while the original Tulu3 mixture achieves an average score of 0.470, decontamination alone shows a negligible impact (0.466). However, adding license filtering reduces average performance by 7.1% (from 0.466 to 0.435), with particularly severe drops on MMLU benchmarks (excluding the MMLU benchmarks almost completely removes the performance gap). Interestingly, some capabilities improve with filtering: TruthfulQA MC2 accuracy increases from 0.486 to 0.518 (+6.6%), and several reasoning tasks show marginal improvements. These results highlight the inherent tension between compliance and model capability, a trade-off we accept as necessary for responsible open-source model development.

We also evaluated the same model configurations on six multilingual benchmarks spanning knowledge (Global-MMLU), mathematical reason-

[11] https://huggingface.co/datasets/allenai/llama-3.1-tulu-3-8b-preference-mixture

Model	Group Averages						
	Knowledge Avg ↑	Commonsense Reasoning Avg ↑	Coding Avg ↑	Math Avg ↑	Reasoning Avg ↑	Instruction Following Avg ↑	Cultural Knowledge Avg ↑
Fully Open Models							
Apertus-70B-Instruct	61.8	66.7	60.0	51.6	55.2	75.0	61.5
Apertus-8B-Instruct	56.4	63.6	51.6	40.4	48.8	70.3	58.6
ALLaM-7B-Instruct-preview	51.2	58.6	47.8	33.8	50.6	59.7	55.2
EuroLLM-22B-Instruct-Preview	57.1	60.5	59.1	49.9	52.0	72.4	57.0
EuroLLM-9B-Instruct	51.6	58.0	53.2	37.7	45.9	62.0	54.3
K2-Chat	55.3	59.8	72.0	53.3	57.1	47.7	56.3
marin-8b-instruct	54.2	55.0	63.5	45.8	51.2	65.4	52.5
Minerva-7B-instruct-v1.0	37.6	47.2	21.1	11.2	31.4	19.6	39.1
OLMo-2-0325-32B-Instruct	67.2	69.5	55.4	57.4	71.0	83.3	58.1
OLMo-2-1124-7B-Instruct	51.4	58.1	48.6	44.4	51.0	65.8	49.7
salamandra-7b-instruct	48.7	58.6	25.3	16.5	39.8	33.7	52.8
SmolLM3-3B	54.3	54.7	71.2	52.1	54.2	71.2	52.7
Teuken-7B-instruct-v0.6	45.8	55.0	35.1	24.0	38.1	30.8	49.7
Open-Weight Models							
gemma-3-12b-it	66.3	49.7	80.0	66.6	72.8	80.1	63.4
gemma-3-27b-it	69.5	52.4	81.0	69.1	75.1	80.6	67.7
Llama-3.1-8B-Instruct	58.8	59.7	73.6	53.2	58.4	75.0	58.2
Llama-3.3-70B-Instruct	71.6	62.0	85.7	68.6	80.8	89.8	69.6
Qwen2.5-72B-Instruct	73.1	60.1	85.0	69.4	76.6	85.0	66.8
Qwen3-32B	67.0	58.4	85.3	71.8	78.4	85.8	65.9
Qwen3-8B	62.0	49.5	81.2	62.5	67.6	84.6	60.4

Table 2: **Post-training Evaluation.** We report average scores across seven benchmark categories: **Knowledge** (MMLU, Global MMLU, TruthfulQA, TruthfulQA Multilingual), **Commonsense Reasoning** (HellaSwag, HellaSwag Multilingual), **Coding** (HumanEval, MBPP), **Math** (GSM8K, MGSM, Hendrycks-Math, MathQA), **Reasoning** (BBH, DROP, ACP-Bool, ACP-MCQ), **Instruction Following** (IFEval, Multi-IF), and **Cultural Knowledge** (INCLUDE, BLEND, CulturalBench, SwitzerlandQA). Higher values indicate better performance. Individual benchmark results are shown in Tables 27-31.

Configuration	MMLU (CoT)	MMLU (CoT-strict)	TruthfulQA MC2	BBH	DROP F1	ACP-Bool	ACP-MCQ	GSM8K	HumanEval Pass@10	MBPP Pass@1	IFEval	ToxiGen	BBQ	Avg.
Tulu3 (original)	**0.542**	**0.513**	0.489	0.482	**0.463**	0.560	0.252	0.482	0.365	0.324	0.536	**0.665**	0.442	**0.470**
+ decontamination	0.538	**0.513**	0.486	0.470	0.461	**0.563**	0.247	0.479	0.353	0.318	**0.547**	0.642	**0.443**	0.466
+ license filtering	0.391	0.253	0.518	**0.490**	0.430	0.551	**0.260**	**0.501**	**0.384**	0.322	0.542	0.598	0.417	0.435

Table 3: **Ablation Study for Decontamination and License Filtering**. Ablation study showing the impact of decontamination and license filtering on Apertus-8B performance across 13 benchmarks. Models were initialized from 10T token checkpoint and finetuned on different data configurations.

ing (MGSM), cultural understanding (INCLUDE, CulturalBench, SwitzerlandQA). As shown in Table 4, the filtering impact on multilingual tasks follows similar patterns to English benchmarks. The original Tulu3 mixture achieves the strongest multilingual performance. Decontamination alone has minimal overall impact, though individual metrics show minor variations. MGSM direct evaluation drops from 0.187 to 0.176 while CulturalBench improves slightly from 0.709 to 0.717. Adding license filtering reduces average performance to 0.456, with MGSM using naive CoT showing the largest relative drop (0.320 → 0.273, -14.7%). Cultural knowledge benchmarks are also affected by filtering, with CulturalBench declining by 5.4%.

Memorization Prevention. To evaluation the impact of Goldfish loss, we evaluate verbatim memorization of training sequences in our long-context pretrained models, Apertus-8B-64k and Apertus-70B-64k. For this evaluation, we injected sequences from the Gutenberg book corpus at different frequencies into our pretraining data (§D.1.4). Subsequently, we prompt the Apertus models with variable-length prefixes from these sequences, and measure the Rouge-L score (Lin, 2004) between generated sequences and the reference continuations from the training data (each of length 500 tokens). In Figure 2, we see that both Apertus-8B and Apertus-70B remain at baseline memorization (Rouge-L $\approx$ 0.18, comparable to Rouge-L be-

Configuration	Global-MMLU	MGSM (Direct)	MGSM (Native CoT)	INCLUDE	CulturalBench	SwitzerlandQA	Avg.
Tulu3 (original)	0.528	0.187	**0.332**	0.509	0.709	**0.592**	**0.476**
+ decontamination	**0.529**	0.176	0.320	**0.510**	**0.717**	0.590	0.474
+ license filtering	0.500	**0.212**	0.273	0.493	0.678	0.579	0.456

Table 4: **Ablation Study for Decontamination and License Filtering on Multilingual Benchmarks**. Apertus-8B was evaluated on global knowledge, mathematical reasoning, and cultural understanding tasks.

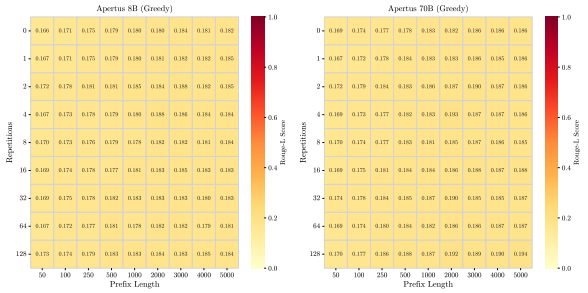

Figure 2: **Apertus is Robust to Verbatim Memorization**. Heatmaps show average Rouge-L scores for suffixes of 500 tokens for Apertus-8B and Apertus-70B. The y-axis represents exposure frequencies during training (1–128), with unexposed bucket 0 serving as a baseline (top row). The prefix length varied from 50 to 5000 tokens. The results demonstrate successful mitigation of verbatim memorization in Apertus, as the Rouge-L scores for both model scales remain at baseline levels.

tween unrelated Gutenberg texts), regardless of the training-time exposure frequency or test-time prefix length of the sequence. We provide more analysis of Apertus memorization capabilities in §F.4.

6 Conclusion

We introduce Apertus, a new model suite designed around three commitments: responsible data practices, global multilingual coverage, and full transparency. Our models are trained on 15T tokens from 1811 languages with retroactive respect for `robots.txt` and related opt-outs, and with a Goldfish-style objective to curb verbatim memorization. We post-train multilingual Apertus-{8B,70B}-Instruct variants to improve interaction across many languages. Our experiments show strong performance across a range of knowledge, cultural, and instruction-following evaluations. While this article only touches on the high-level final design of Apertus, we provide more comprehensive development details in the Appendix.

Limitations

While Apertus establishes a baseline for compliance and transparency, the current release has several limitations that represent clear avenues for future work:

- **Scaling.** Train larger models and longer-context variants while preserving the compliance and transparency guarantees established.
- **Distillation.** Distill the 70B model into smaller students for constrained settings without eroding multilingual and safety properties.
- **Reasoning with adaptive compute.** Explore test-time variable computation that allocates more steps to harder inputs, including internal chain-of-thought tokens, routing, and variable-depth architectures (Wei et al., 2022).
- **RL with verifiers.** Develop RLVR pipelines that combine preference optimization with explicit verifiers for math, code, and other tasks with verifiable reasoning steps (OpenAI, 2024; Guo et al., 2025).
- **Multimodality.** Extend the stack to visual, audio, and other data modalities while maintaining the same compliance standards for data collection and release.
- **Societal alignment.** Elicit and model diverse multilingual preferences to inform alignment objectives and evaluation (Stammbach et al., 2024; Kirk et al., 2025).
- **Field evaluation.** Run structured studies with domain professionals and the public to assess reliability, usability, and real-world impact across languages and sectors.

We anticipate addressing many of these limitations in subsequent versions of Apertus.

References

2024. Scaling neural machine translation to 200 languages. *Nature*, 630(8018):841–846.

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *Preprint*, arXiv:2305.13245.

Mehdi Ali, Michael Fromm, Klaudia Thellmann, Jan Ebert, Alexander Arno Weber, Richard Rutmann, Charvi Jain, Max Lübbering, Daniel Steinigen, Johannes Leveling, and 1 others. 2024. Teuken-7b-base & teuken-7b-instruct: Towards european llms. *arXiv preprint arXiv:2410.03730*.

Loubna Ben allal, Anton Lozhkov, Elie Bakouch, Gabriel Martin Blazquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Agustín Piqueres Lajarín, Hynek Kydlíček, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan Son NGUYEN, Ben Burtenshaw, Clémentine Fourrier, Haojun Zhao, Hugo Larcher, Mathieu Morlon, Cyril Zakka, and 3 others. 2025. SmolLM2: When smol goes big — data-centric training of a fully open small language model. In *Second Conference on Language Modeling*.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku.

Catherine Arnett, Eliot Jones, Ivan P. Yamshchikov, and Pierre-Carl Langlais. 2024. Toxicity of the commons: Curating open-source pre-training data. *Preprint*, arXiv:2410.22587.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *Preprint*, arXiv:1607.06450.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenhang Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, K. Lu, and 31 others. 2023. Qwen technical report. *ArXiv*, abs/2309.16609.

Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, and 1 others. 2025. Kimi K2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, and 12 others. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *CoRR*, abs/2204.05862.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, and 32 others. 2022b. Constitutional AI: harmlessness from AI feedback. *CoRR*, abs/2212.08073.

Elie Bakouch, Loubna Ben Allal, Anton Lozhkov, Nouamane Tazi, Lewis Tunstall, Carlos Miguel Patiño, Edward Beeching, Aymeric Roucher, Aksel Joonas Reedi, Quentin Gallouédec, Kashif Rasul, Nathan Habib, Clémentine Fourrier, Hynek Kydlicek, Guilherme Penedo, Hugo Larcher, Mathieu Morlon, Vaibhav Srivastav, Joshua Lochner, and 4 others. 2025. SmolLM3: smol, multilingual, long-context reasoner. `https://huggingface.co/blog/smollm3`.

M Saiful Bari, Yazeed Alnumay, Norah A. Alzahrani, Nouf M. Alotaibi, Hisham Abdullah Alyahya, Sultan AlRashed, Faisal Abdulrahman Mirza, Shaykhah Z. Alsubaie, Hassan A. Alahmed, Ghadah Alabduljabbar, Raghad Alkhathran, Yousef Almushayqih, Raneem Alnajim, Salman Alsubaihi, Maryam Al Mansour, Saad Amin Hassan, Dr. Majed Alrubaian, Ali Alammari, Zaki Alawami, and 7 others. 2025. ALLam: Large language models for arabic and english. In *The Thirteenth International Conference on Learning Representations*.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Sid Black, Gao Leo, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow. If you use this software, please cite it using these metadata.

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, virtual+Dublin. Association for Computational Linguistics.

Cody Blakeney, Mansheej Paul, Brett W. Larsen, Sean Owen, and Jonathan Frankle. 2024. Does your data

spark joy? performance gains from domain upsampling at the end of training. In *Workshop on Efficient Systems for Foundation Models II @ ICML2024*.

Kaj Bostrom and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624, Online. Association for Computational Linguistics.

Blanca Calvo Figueras, Eneko Sagarzazu, Julen Etxaniz, Jeremy Barnes, Pablo Gamallo, Iria de Dios-Flores, and Rodrigo Agerri. 2025. Truth knows no language: Evaluating truthfulness beyond English. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 31204–31218, Vienna, Austria. Association for Computational Linguistics.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: evaluating and testing unintended memorization in neural networks. In *Proceedings of the 28th USENIX Conference on Security Symposium*, SEC'19, page 267–284, USA. USENIX Association.

Yekun Chai, Yewei Fang, Qiwei Peng, and Xuhong Li. 2024. Tokenization falling short: On subword robustness in large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1582–1599, Miami, Florida, USA. Association for Computational Linguistics.

Kent Chang, Mackenzie Cramer, Sandeep Soni, and David Bamman. 2023. Speak, memory: An archaeology of books known to ChatGPT/GPT-4. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7312–7327, Singapore. Association for Computational Linguistics.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Yu Ying Chiu, Liwei Jiang, Bill Yuchen Lin, Chan Young Park, Shuyue Stella Li, Sahithya Ravi, Mehar Bhatia, Maria Antoniak, Yulia Tsvetkov, Vered Shwartz, and Yejin Choi. 2025. CulturalBench: A robust, diverse and challenging benchmark for measuring LMs' cultural knowledge through human-AI red-teaming. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 25663–25701, Vienna, Austria. Association for Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and 1 others. 2023. Palm: Scaling language modeling with pathways. *Journal of machine learning research*, 24(240):1–113.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *ArXiv*, abs/1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Marin Community. 2025. Marin 8b instruct. `https://huggingface.co/marin-community/marin-8b-instruct`. Accessed: 2025-09-01.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Felipe A. Cruz and Alberto Madonna. 2024. Containers-first user environments on hpe cray ex. In *Proceedings of the Cray User Group Conference (CUG 2024). Cray User Group.(May 2024)*.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2024. Ultrafeedback: boosting language models with scaled ai feedback. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, and 1 others. 2023. Scaling vision transformers to 22 billion parameters. In *International conference on machine learning*, pages 7480–7512. PMLR.

Yue Deng, Wenxuan Zhang, Sinno Jialin Pan, and Lidong Bing. 2024. Multilingual jailbreak challenges in large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Daniel Deutsch, Eleftheria Briakou, Isaac Rayburn Caswell, Mara Finkelstein, Rebecca Galor, Juraj Juraska, Geza Kovacs, Alison Lui, Ricardo Rei, Jason Riesa, Shruti Rijhwani, Parker Riley, Elizabeth Salesky, Firas Trabelsi, Stephanie Winkler, Biao Zhang, and Markus Freitag. 2025. WMT24++: Expanding the language coverage of WMT24 to 55 languages & dialects. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 12257–12284, Vienna, Austria. Association for Computational Linguistics.

Xin Dong, Yonggan Fu, Shizhe Diao, Wonmin Byeon, ZIJIA CHEN, Ameya Sunil Mahabaleshwarkar, Shih-Yang Liu, Matthijs Van keirsbilck, Min-Hung Chen, Yoshi Suhara, Yingyan Celine Lin, Jan Kautz, and Pavlo Molchanov. 2025. Hymba: A hybrid-head architecture for small language models. In *The Thirteenth International Conference on Learning Representations*.

Aleksandr Dremov, Alexander Hägele, Atli Kosson, and Martin Jaggi. 2025. Training dynamics of the cooldown stage in warmup-stable-decay learning rate scheduler. *Transactions on Machine Learning Research*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Antoine Dussolle, A. Cardeña, Shota Sato, and Peter Devine. 2025. M-IFEval: Multilingual instruction-following evaluation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 6161–6176, Albuquerque, New Mexico. Association for Computational Linguistics.

Dongyang Fan, Vinko Sabolčec, Matin Ansaripour, Ayush Kumar Tarun, Martin Jaggi, Antoine Bosselut, and Imanol Schlag. 2025. Can performant LLMs be ethical? quantifying the impact of web crawling opt-outs. In *Second Conference on Language Modeling*.

Igor Fedorov, Kate Plawiak, Lemeng Wu, Tarek Elgamal, Naveen Suda, Eric Smith, Hongyuan Zhan, Jianfeng Chi, Yuriy Hulovatyy, Kimish Patel, Zechun Liu, Changsheng Zhao, Yangyang Shi, Tijmen Blankevoort, Mahesh Pasupuleti, Bilge Soran, Zacharie Delpierre Coudert, Rachad Alao, Raghuraman Krishnamoorthi, and Vikas Chandra. 2024. Llama guard 3-1b-int4: Compact and efficient safeguard for human-ai conversations. *CoRR*, abs/2411.17713.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020. CodeBERT: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1536–1547, Online. Association for Computational Linguistics.

Negar Foroutan, Clara Meister, Debjit Paul, Joel Niklaus, Sina Ahmadi, Antoine Bosselut, and Rico Sennrich. 2025. Parity-aware byte-pair encoding: Improving cross-lingual fairness in tokenization. In *arXiv*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. The language model evaluation harness.

Tao Ge, Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. 2025. Scaling synthetic data creation with 1,000,000,000 personas. *Preprint*, arXiv:2406.20094.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 3356–3369. Association for Computational Linguistics.

Josh A. Goldstein, Girish Sastry, Micah Musser, Renee DiResta, Matthew Gentzel, and Katerina Sedova. 2023. Generative language models and automated influence operations: Emerging threats and potential mitigations. *ArXiv*, abs/2301.04246.

Aitor Gonzalez-Agirre, Marc Pàmies, Joan Llop, Irene Baucells, Severino Da Dalt, Daniel Tamayo, José Javier Saiz, Ferran Espuña, Jaume Prats, Javier Aula-Blasco, and 1 others. 2025. Salamandra technical report. *arXiv preprint arXiv:2502.08489*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Dirk Groeneveld, Iz Beltagy, Evan Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, and 24 others. 2024. OLMo: Accelerating the science of language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15789–15809, Bangkok, Thailand. Association for Computational Linguistics.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Alexander Hägele, Elie Bakouch, Atli Kosson, Loubna Ben Allal, Leandro Von Werra, and Martin

Jaggi. 2024. Scaling Laws and Compute-Optimal Training Beyond Fixed Training Durations. *Advances in Neural Information Processing Systems*.

Abhimanyu Hans, John Kirchenbauer, Yuxin Wen, Neel Jain, Hamid Kazemi, Prajwal Singhania, Siddharth Singh, Gowthami Somepalli, Jonas Geiping, Abhinav Bhatele, and Tom Goldstein. 2024. Be like a goldfish, don't memorize! mitigating memorization in generative LLMs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. Toxigen: A large-scale machine-generated dataset for implicit and adversarial hate speech detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. 2020. Query-key normalization for transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4246–4253, Online. Association for Computational Linguistics.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. 2024. RULER: What's the real context size of your long-context language models? In *First Conference on Language Modeling*.

Shengding Hu, Yuge Tu, Xu Han, Ganqu Cui, Chaoqun He, Weilin Zhao, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Xinrong Zhang, Zhen Leng Thai, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, and 5 others. 2024. MiniCPM: Unveiling the potential of small language models with scalable training strategies. In *First Conference on Language Modeling*.

Allen Hao Huang and Imanol Schlag. 2025. Deriving activation functions using integration. *Preprint*, arXiv:2411.13010.

Jie Huang, Hanyin Shao, and Kevin Chen-Chuan Chang. 2022. Are large pre-trained language models leaking your personal information? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2038–2047, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

HuggingFaceTB. 2025. Smollm3-3b. `https://huggingface.co/HuggingFaceTB/SmolLM3-3B`. Accessed: 2025-09-01.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. 2023. Llama guard: Llm-based input-output safeguard for human-ai conversations. *CoRR*, abs/2312.06674.

Devansh Jain, Priyanshu Kumar, Samuel Gehman, Xuhui Zhou, Thomas Hartvigsen, and Maarten Sap. 2024. Polyglotoxicityprompts: Multilingual evaluation of neural toxic degeneration in large language models. In *First Conference on Language Modeling*.

Aiqi Jiang, Xiaohan Yang, Yang Liu, and Arkaitz Zubiaga. 2021. Swsr: A chinese dataset and lexicon for online sexism detection. *Preprint*, arXiv:2108.03070.

Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *ArXiv*, abs/2310.06825.

Gemma Team Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ram'e, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean-Bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, Gael Liu, and 191 others. 2025. Gemma 3 technical report. *ArXiv*, abs/2503.19786.

Nikhil Kandpal, Brian Lester, Colin Raffel, Sebastian Majstorovic, Stella Biderman, Baber Abbasi, Luca Soldaini, Enrico Shippole, A. Feder Cooper, Aviya Skowron, Shayne Longpre, Lintang Sutawika, Alon Albalak, Zhenlin Xu, Guilherme Penedo, Loubna Ben allal, Elie Bakouch, John David Pressman, Honglu Fan, and 8 others. 2025. The common pile v0.1: An 8TB dataset of public domain and openly licensed text. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Antonia Karamolegkou, Jiaang Li, Li Zhou, and Anders Søgaard. 2023. Copyright violations and large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7403–7412, Singapore. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Hannah Rose Kirk, Alexander Whitefield, Paul Röttger, Andrew Bean, Katerina Margatina, Juan Ciro, Rafael Mosquera, Max Bartolo, Adina Williams, He He, Bertie Vidgen, and Scott A. Hale. 2025. The prism

alignment dataset: what participatory, representative and individualised human feedback reveals about the subjective and multicultural alignment of large language models. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA. Curran Associates Inc.

Denis Kocetkov, Raymond Li, Loubna Ben allal, Jia LI, Chenghao Mou, Yacine Jernite, Margaret Mitchell, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, Dzmitry Bahdanau, Leandro Von Werra, and Harm de Vries. 2023. The stack: 3 TB of permissively licensed source code. *Transactions on Machine Learning Research*.

Tom Kocmi, Eleftherios Avramidis, Rachel Bawden, Ondřej Bojar, Anton Dvorkovich, Christian Federmann, Mark Fishel, Markus Freitag, Thamme Gowda, Roman Grundkiewicz, Barry Haddow, Marzena Karpinska, Philipp Koehn, Benjamin Marie, Christof Monz, Kenton Murray, Masaaki Nagata, Martin Popel, Maja Popović, and 3 others. 2024. Findings of the WMT24 general machine translation shared task: The LLM era is here but MT is not solved yet. In *Proceedings of the Ninth Conference on Machine Translation*, pages 1–46, Miami, Florida, USA. Association for Computational Linguistics.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand.

Harsha Kokel, Michael Katz, Kavitha Srinivas, and Shirin Sohrabi. 2025. Acpbench: Reasoning about action, change, and planning. In *AAAI*. AAAI Press.

Andrei Kucharavy, Zachary Schillaci, Loïc Maréchal, Maxime Würsch, Ljiljana Dolamic, Remi Sabonnadiere, Dimitri Percia David, Alain Mermoud, and Vincent Lenders. 2023. Fundamentals of generative large language models and perspectives in cyber-defense. *CoRR*, abs/2303.12132.

Viet Lai, Chien Nguyen, Nghia Ngo, Thuat Nguyen, Franck Dernoncourt, Ryan Rossi, and Thien Nguyen. 2023. Okapi: Instruction-tuned large language models in multiple languages with reinforcement learning from human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 318–327, Singapore. Association for Computational Linguistics.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Christopher Wilhelm, Luca Soldaini, and 4 others. 2025. Tulu 3: Pushing frontiers in open language model post-training. In *Second Conference on Language Modeling*.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, and 1 others. 2024. Datacomp-lm: In search of the next generation of training sets for language models. *Advances in Neural Information Processing Systems*, 37:14200–14282.

Raymond Li, Loubna Ben allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia LI, Jenny Chim, Qian Liu, Evgenii Zheltonozhskii, Terry Yue Zhuo, Thomas Wang, Olivier Dehaene, Joel Lamy-Poirier, Joao Monteiro, Nicolas Gontier, Ming-Ho Yee, and 39 others. 2023. Starcoder: may the source be with you! *Transactions on Machine Learning Research*. Reproducibility Certification.

Hauke Licht, Rupak Sarkar, Patrick Y. Wu, Pranav Goel, Niklas Stoehr, Elliott Ash, and Alexander Hoyle. 2025. Measuring scalar constructs in social science with LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Chris Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, and Yang Liu. 2026. Skywork-reward-v2: Scaling preference data curation via human-AI synergy. In *The Fourteenth International Conference on Learning Representations*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2021. Logiqa: a challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3622–3628.

Zhengzhong Liu, Aurick Qiao, Willie Neiswanger, Hongyi Wang, Bowen Tan, Tianhua Tao, Junbo Li, Yuqi Wang, Suqi Sun, Omkar Pangarkar, Richard Fan, Yi Gu, Victor Miller, Yonghao Zhuang, Guowei He, Haonan Li, Fajri Koto, Liping Tang, Nikhil Ranjan, and 8 others. 2024b. LLM360: Towards fully transparent open-source LLMs. In *First Conference on Language Modeling*.

Zhengzhong Liu, Bowen Tan, Hongyi Wang, Willie Neiswanger, Tianhua Tao, Haonan Li, Fajri Koto, Yuqi Wang, Suqi Sun, Omkar Pangarkar, Richard Fan, Yi Gu, Victor Miller, Liqun Ma, Liping Tang, Nikhil Ranjan, Yonghao Zhuang, Guowei He, Renxi Wang, and 6 others. 2025a. Llm360 k2: Building a 65b 360-open-source large language model from scratch. *ArXiv*, abs/2501.07124.

Zhengzhong Liu, Bowen Tan, Hongyi Wang, Willie Neiswanger, Tianhua Tao, Haonan Li, Fajri Koto, Yuqi Wang, Suqi Sun, Omkar Pangarkar, and 1 others. 2025b. Llm360 k2: Building a 65b 360-open-source large language model from scratch. *arXiv preprint arXiv:2501.07124*.

Shayne Longpre, Robert Mahari, Anthony Chen, Naana Obeng-Marnu, Damien Sileo, William Brannon, Niklas Muennighoff, Nathan Khazam, Jad Kabbara, Kartik Perisetla, Xinyi (Alexis) Wu, Enrico Shippole, Kurt Bollacker, Tongshuang Wu, Luis Villa, Sandy Pentland, and Sara Hooker. 2024a. A large-scale audit of dataset licensing and attribution in ai. *Nature Machine Intelligence*, 6(8):975–987.

Shayne Longpre, Robert Mahari, Ariel Lee, Campbell Lund, Hamidah Oderinwale, William Brannon, Nayan Saxena, Naana Obeng-Marnu, Tobin South, Cole Hunter, and 1 others. 2024b. Consent in crisis: The rapid decline of the ai data commons. *Advances in Neural Information Processing Systems*, 37:108042–108087.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *ArXiv*, abs/1711.05101.

Risto Luukkonen, Jonathan Burdge, Elaine Zosa, Aarne Talman, Ville Komulainen, Väinö Hatanpää, Peter Sarlin, and Sampo Pyysalo. 2025. Poro 34B and the blessing of multilinguality. In *Proceedings of the Joint 25th Nordic Conference on Computational Linguistics and 11th Baltic Conference on Human Language Technologies (NoDaLiDa/Baltic-HLT 2025)*, pages 367–382, Tallinn, Estonia. University of Tartu Library.

Subhabrata Majumdar and Terry Vogelsang. 2024. *Towards Safe LLMs Integration*, pages 243–247. Springer Nature Switzerland, Cham.

Maxime Martinasso, Mark Klein, and Thomas Schulthess. 2025. Alps, a versatile research infrastructure. In *Proceedings of the Cray User Group*, pages 156–165.

Pedro Henrique Martins, João Alves, Patrick Fernandes, , Nuno M. Guerreiro, Ricardo Rei, Amin Farajian, Mateusz Klimaszewski, Duarte M. Alves, José Pombal, Manuel Faysse, Pierre Colombo, François Yvon, Barry Haddow, José G. C. de Souza, Alexandra Birch, and André F. T. Martins. 2025. Eurollm-9b: Technical report.

PH Martins, P Fernandes, J Alves, NM Guerreiro, R Rei, DM Alves, J Pombal, A Farajian, M Faysse, M Klimaszewski, and 1 others. 2024. Eurollm: Multilingual language models for europe (arxiv: 2409.16235). arxiv.

Simon Matrenok, Skander Moalla, and Caglar Gulcehre. 2025. Quantile reward policy optimization: Alignment with pointwise regression and exact partition functions. *Preprint*, arXiv:2507.08068.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, and 1 others. 2024. Harmbench: a standardized evaluation framework for automated red teaming and robust refusal. In *Proceedings of the 41st International Conference on Machine Learning*, pages 35181–35224.

Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. 2018. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*.

Clara Meister. 2025. TokEval: A tokenizer analysis suite.

William Merrill, Shane Arora, Dirk Groeneveld, and Hannaneh Hajishirzi. Critical batch size revisited: A simple empirical approach to large-batch language model training. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Gemma Team Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, L. Sifre, Morgane Rivière, Mihir Kale, J Christopher Love, Pouya Dehghani Tafti, L'eonard Hussenot, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Am'elie H'eliou, Andrea Tacchetti, and 88 others. 2024. Gemma: Open models based on gemini research and technology. *ArXiv*, abs/2403.08295.

Bettina Messmer, Vinko Sabolčec, and Martin Jaggi. 2025. Enhancing multilingual LLM pretraining with model-based data selection. In *Proceedings of the 10th edition of the Swiss Text Analytics Conference*, pages 31–56, Winterthur, Switzerland. Association for Computational Linguistics.

Meta AI. 2025. Introducing lLama-4: Advancing multimodal intelligence. `https://ai.meta.com/blog/llama-4-multimodal-intelligence/`. Accessed: 2025-09-01.

Skander Moalla. 2025. Python Machine Learning Research Template.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20275–20321, Suzhou, China. Association for Computational Linguistics.

Junho Myung, Nayeon Lee, Yi Zhou, Jiho Jin, Rifki Afina Putri, Dimosthenis Antypas, Hsuvas Borkakoty, Eunsu Kim, Carla Perez-Almendros, Abinew Ali Ayele, Victor Gutierrez Basulto, Yazmin Ibanez-Garcia, Hwaran Lee, Shamsuddeen Hassan Muhammad, Kiwoong Park, Anar Sabuhi Rzayev, Nina White, Seid Muhie Yimam, Mohammad Taher Pilehvar, and 3 others. 2024. BLEnd: A benchmark for LLMs on everyday knowledge in diverse cultures and languages. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Milad Nasr, Javier Rando, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A. Feder Cooper, Daphne Ippolito, Christopher A. Choquette-Choo, Florian Tramèr, and Katherine Lee. 2025. Scalable extraction of training data from aligned, production language models. In *The Thirteenth International Conference on Learning Representations*.

Zhiyuan Ning, Tianle Gu, Jiaxin Song, Shixin Hong, Lingyu Li, Huacan Liu, Jie Li, Yixu Wang, Meng Lingyu, Yan Teng, and 1 others. 2025. Linguasafe: A comprehensive multilingual safety benchmark for large language models. *arXiv preprint arXiv:2508.12733*.

Sapienza NLP. 2024. Minerva-7b-instruct-v1.0. `https://huggingface.co/sapienzanlp/Minerva-7B-instruct-v1.0`. Accessed: 2025-09-01.

OpenAI, :, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Haiming Bao, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sebastien Bubeck, and 108 others. 2025. gpt-oss-120b & gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

OpenAI. 2023. Gpt-4 technical report. *CoRR*, abs/2303.08774.

OpenAI. 2024. Openai o1 system card.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Matteo Pagliardini, Pierre Ablin, and David Grangier. 2025. The adEMAMix optimizer: Better, faster, older. In *The Thirteenth International Conference on Learning Representations*.

Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. 2024. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *Preprint*, arXiv:2402.13228.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Kanghee Park, Jiayu Wang, Taylor Berg-Kirkpatrick, Nadia Polikarpova, and Loris D'Antoni. 2024. Grammar-aligned decoding. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Alicia Parrish, Angelica Chen, Nikita Nangia, Vishakh Padmakumar, Jason Phang, Jana Thompson, Phu Mon Htut, and Samuel Bowman. 2022. BBQ: A hand-built bias benchmark for question answering. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2086–2105, Dublin, Ireland. Association for Computational Linguistics.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, Thomas Wolf, and 1 others. 2024a. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849.

Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro Von Werra, and Thomas Wolf. 2025. Fineweb2: One pipeline to scale them all — adapting pre-training data processing to every language. In *Second Conference on Language Modeling*.

Guilherme Penedo, Hynek Kydlíček, Alessandro Cappelli, Mario Sasko, and Thomas Wolf. 2024b. Datatrove: large scale data processing.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2024. YaRN: Efficient context window extension of large language models. In *The Twelfth International Conference on Learning Representations*.

Edoardo Maria Ponti, Goran Glavaš, Olga Majewska, Qianchu Liu, Ivan Vulić, and Anna Korhonen. 2020. XCOPA: A multilingual dataset for causal commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2362–2376, Online. Association for Computational Linguistics.

A Yang Qwen, Baosong Yang, B Zhang, B Hui, B Zheng, B Yu, Chengpeng Li, D Liu, F Huang, H Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, and 1 others. 2018. Improving language understanding by generative pre-training.

Rafael Rafailov, Yaswanth Chittepu, Ryan Park, Harshit Sushil Sikchi, Joey Hejna, Brad Knox, Chelsea Finn, and Scott Niekum. 2024. Scaling laws for reward model overoptimization in direct alignment algorithms. *Advances in Neural Information Processing Systems*, 37:126207–126242.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: your language model is secretly a reward model. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Gemma Team Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, L'eonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ram'e, Johan Ferret, Peter Liu, Pouya Dehghani Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, and 176 others. 2024. Gemma 2: Improving open language models at a practical size. *ArXiv*, abs/2408.00118.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Angelika Romanou, Negar Foroutan, Anna Sotnikova, Zeming Chen, Sree Harsha Nelaturu, Shivalika Singh, Rishabh Maheshwary, Micol Altomare, Mohamed A Haggag, Alfonso Amayuelas, Azril Hafizi Amirudin, Viraat Aryabumi, Danylo Boiko, Michael Chang, Jenny Chim, Gal Cohen, Kumar Dalmia, Abraham Diress, Sharad Duwal, and 39 others. 2025. Include: Evaluating multilingual language understanding with regional knowledge. In *ICLR*.

Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. 2024. XSTest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5377–5400, Mexico City, Mexico. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Mansi Sakarvadia, Aswathy Ajith, Arham Mushtaq Khan, Nathaniel C Hudson, Caleb Geniesse, Kyle Chard, Yaoqing Yang, Ian Foster, and Michael W. Mahoney. 2025. Mitigating memorization in language models. In *The Thirteenth International Conference on Learning Representations*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ili'c, Daniel Hesslow, Roman Castagn'e, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, and 372 others. 2022. Bloom: A 176b-parameter open-access multilingual language model. *ArXiv*, abs/2211.05100.

Fabian Schaipp, Alexander Hägele, Adrien Taylor, Umut Simsekli, and Francis Bach. 2025. The surprising agreement between convex optimization theory and learning-rate scheduling for large model training. *Forty-second International Conference on Machine Learning*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *Preprint*, arXiv:1707.06347.

Stefano Schuppli, Fawzi Mohamed, Henrique Mendonca, Nina Mujkanovic, Elia Palme, Dino Conciatore, Lukas Drescher, Miguel Gila, Pim Witlox, Joost VandeVondele, and 1 others. 2025. Evolving hpc services to enable ml workloads on hpe cray ex. In *Proceedings of the Cray User Group*, pages 166–177.

Andrei Semenov, Matteo Pagliardini, and Martin Jaggi. 2025. Benchmarking optimizers for large language model pretraining. *arXiv preprint arXiv:2509.01440*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. 2023. Language models are multilingual chain-of-thought reasoners. In *The Eleventh International Conference on Learning Representations*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *Preprint*, arXiv:1909.08053.

Shivalika Singh, Angelika Romanou, Clémentine Fourrier, David Ifeoluwa Adelani, Jian Gang Ngui, Daniel Vila-Suero, Peerat Limkonchotiwat, Kelly Marchisio, Wei Qi Leong, Yosephine Susanto, Raymond Ng, Shayne Longpre, Sebastian Ruder, Wei-Yin Ko, Antoine Bosselut, Alice Oh, Andre Martins, Leshem Choshen, Daphne Ippolito, and 4 others. 2025. Global MMLU: Understanding and addressing cultural and linguistic biases in multilingual evaluation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18761–18799, Vienna, Austria. Association for Computational Linguistics.

Samuel L Smith, Pieter-Jan Kindermans, Chris Ying, and Quoc V Le. 2018. Don't decay the learning rate, increase the batch size. In *International Conference on Learning Representations*.

David So, Wojciech Mańke, Hanxiao Liu, Zihang Dai, Noam Shazeer, and Quoc V Le. 2021. Searching for efficient transformers for language modeling. In *Advances in Neural Information Processing Systems*.

Dominik Stammbach, Philine Widmer, Eunjung Cho, Çağlar Gùlçehre, and Elliott Ash. 2024. Aligning large language models with diverse political viewpoints. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7257–7267.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. 2023. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, Toronto, Canada. Association for Computational Linguistics.

Perplexity AI AI Team. 2025. open-sourcing R1 1776. https://web.archive.org/web/20250816143635/https://www.perplexity.ai/hub/blog/open-sourcing-r1-1776. Accessed: 2025-08-29.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aur'elien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Niko lay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cris tian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023b. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Ahmet Üstün, Viraat Aryabumi, Zheng Yong, Wei-Yin Ko, Daniel D'souza, Gbemileke Onilude, Neel Bhandari, Shivalika Singh, Hui-Lee Ooi, Amr Kayid, Freddie Vargus, Phil Blunsom, Shayne Longpre, Niklas Muennighoff, Marzieh Fadaee, Julia Kreutzer, and Sara Hooker. 2024. Aya model: An instruction finetuned open-access multilingual language model. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15894–15939, Bangkok, Thailand. Association for Computational Linguistics.

Jannis Vamvas, Ignacio Pérez Prat, Not Soliva, Sandra Baltermia-Guetg, Andrina Beeli, Simona Beeli, Madlaina Capeder, Laura Decurtins, Gian Peder Gregori, Flavia Hobi, Gabriela Holderegger, Arina Lazzarini, Viviana Lazzarini, Walter Rosselli, Bettina Vital, Anna Rutkiewicz, and Rico Sennrich. 2025. Expanding the WMT24++ benchmark with Rumantsch Grischun, Sursilvan, Sutsilvan, Surmiran, Puter, and Vallader. In *Proceedings of the Tenth Conference on Machine Translation*, pages 1028–1047, Suzhou, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Evan Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Allyson Ettinger, and 23 others. 2025. 2 OLMo 2 furious (COLM's version). In *Second Conference on Language Modeling*.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Victor Wang, Michael JQ Zhang, and Eunsol Choi. 2025. Improving LLM-as-a-judge inference with the judgment distribution. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 23173–23199, Suzhou, China. Association for Computational Linguistics.

Wenxuan Wang, Zhaopeng Tu, Chang Chen, Youliang Yuan, Jen-tse Huang, Wenxiang Jiao, and Michael Lyu. 2024. All languages matter: On the multilingual safety of LLMs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5865–5877, Bangkok, Thailand. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances*

in neural information processing systems, 35:24824–24837.

Rachel Wicks, Matt Post, and Philipp Koehn. 2024. Recovering document annotations for sentence-level bitext. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9876–9890, Bangkok, Thailand. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. 2020. On layer normalization in the transformer architecture. In *International conference on machine learning*, pages 10524–10533. PMLR.

Yixuan Xu. 2025. Quantifying training data retention in large language models: An analysis of pretraining factors and mitigation strategies.

Yixuan Xu, Antoine Bosselut, and Imanol Schlag. 2025. Positional fragility in LLMs: How offset effects reshape our understanding of memorization risks. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *ArXiv*, abs/2505.09388.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 39 others. 2024a. Qwen2 technical report. *ArXiv*, abs/2407.10671.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, and 25 others. 2024b. Qwen2.5 technical report. *ArXiv*, abs/2412.15115.

Zheng Xin Yong, Cristina Menghini, and Stephen Bach. 2023. Low-resource languages jailbreak GPT-4. In *Socially Responsible Language Modelling Research*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. 2022. Scaling vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12104–12113.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. *Advances in neural information processing systems*, 32.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068.

Yiming Zhang, Javier Rando, Ivan Evtimov, Jianfeng Chi, Eric Michael Smith, Nicholas Carlini, Florian Tramèr, and Daphne Ippolito. 2025. Persistent pre-training poisoning of LLMs. In *The Thirteenth International Conference on Learning Representations*.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2024. AGIEval: A human-centric benchmark for evaluating foundation models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2299–2314, Mexico City, Mexico. Association for Computational Linguistics.

Enyu Zhou, Guodong Zheng, Binghai Wang, Zhiheng Xi, Shihan Dou, Rong Bao, Wei Shen, Limao Xiong, Jessica Fan, Yurong Mou, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. 2025a. RMB: Comprehensively benchmarking reward models in LLM alignment. In *The Thirteenth International Conference on Learning Representations*.

Fan Zhou, Zengzhi Wang, Nikhil Ranjan, Zhoujun Cheng, Liping Tang, Guowei He, Zhengzhong Liu, and Eric P. Xing. 2025b. Megamath: Pushing the limits of open math corpora. In *Second Conference on Language Modeling*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

Thomas P Zollo, Andrew Wei Tung Siah, Naimeng Ye, Ang Li, and Hongseok Namkoong. 2025. Personal-LLM: Tailoring LLMs to individual preferences. In

The Thirteenth International Conference on Learning Representations.

Appendix

Contents

A Safety Advisory Statement

The Apertus models, while trained at large scale and demonstrating general purpose capabilities, have limitations that must be considered before deploying for real-world use. First, while these models have been tested on a variety of safety benchmarks and environments, they may still produce hallucinations, degenerate as they produce text, generate toxic outputs, and manifest other unsafe behaviors. Second, these models are language-only, only capable of processing text, and cannot process other modalities (such as images). Apertus should only be deployed after extensive use-case alignment and additional testing.

B Related Work

LLMs. Large language models (LLMs) have demonstrated remarkable performance across a wide range of tasks (OpenAI, 2023; Kamath et al., 2025; Anthropic, 2024). Their ability to generalize across diverse domains has established them as foundational tools in both research and industry.

Open-Weight Models. With the strong performance of proprietary models such as GPT, Gemini, and Claude, various open-weight alternatives have emerged, enabling research on large-scale language models. Notable examples include LLaMA, DeepSeek, Qwen, and the recently released GPT-OSS (Inan et al., 2023; Touvron et al., 2023b; Grattafiori et al., 2024; Meta AI, 2025; Bai et al., 2023; Yang et al., 2024a,b, 2025; OpenAI et al., 2025). Despite their accessibility, these models often lack transparency regarding training methodologies and data sources, making it difficult to scrutinize them for biases or to verify compliance with data usage regulations.

Fully Open Models. Fully open models provide access to training data and code in addition to model weights. Early initiatives in this space include the GPT-Neo family and BLOOM (Black et al., 2021; Wang and Komatsuzaki, 2021; Black et al., 2022; Scao et al., 2022). More recently, fully open models have received increasing attention, with releases such as SmolLM2, OLMo, and OLMo 2 (allal et al., 2025; Groeneveld et al., 2024; Walsh et al., 2025) focusing on English capabilities. Efforts such as EuroLLM (Martins et al., 2024) and SmolLM3 (Bakouch et al., 2025) emphasize multilinguality. The Apertus models extend this line of work by incorporating data compliance considerations and supporting 1800 languages.

C Model Architecture & Pretraining

This section details the architecture and pretraining recipe for the Apertus suite of pretrained models. Key choices include the use of a new xIELU activation function, the AdEMAMix optimizer, QK-Norm, Pre-Norm, and Goldfish loss for memorization mitigation. We first provide an overview of the architecture design (Appendix C.1), tokenizer (Appendix C.2) and the algorithms for the main pretraining stage (Appendix C.3). We then describe the ablation studies behind our design choices in Appendix C.5, where experiments with our architecture and optimization setup improve efficiency by 30–40% both at 1B and 3B scale and in a short replication of OLMo2 (1B and 7B). This is followed by the details of the long-context extension in Appendix C.6. Finally, we provide a retrospective of the final training, designs that did not make it into this version, and future directions in Appendix C.7.

Codebase. The pretraining codebase is built on NVIDIA's Megatron-LM (Shoeybi et al., 2019). We extend the codebase with multiple functionalities (*e.g.*, dataloader format, logging during training) and necessary modifications for our architecture (activation function, loss, optimizer). More details on efficiency, scaling, and infrastructure are provided in Section G.

C.1 Model Architecture

Overview. The Apertus architecture is a dense decoder-only Transformer (Vaswani et al., 2017; Radford et al., 2018). The basic architecture consists of a deep stack of Transformer blocks. Each block contains a multi-head self-attention mechanism, followed by a feed-forward network (MLP), with residual connections and normalization applied around each sublayer. We adapt this architecture across two scales:

- Apertus 8B, with 32 layers and 32 parallel attention heads.
- Apertus 70B, with 80 layers and 64 parallel attention heads.

The main characteristics and hyperparameters of the models are listed in Table 5. Besides established modifications to the original Transformer, such as grouped-query attention (GQA), RoPE, and RMSNorm, we improve the architecture efficiency through the use of QK-Norms (Henry et al., 2020; Dehghani et al., 2023) and the activation function xIELU (Huang and Schlag, 2025). The following list describes each modification in more detail.

No biases. We remove all bias terms from the architecture (Chowdhery et al., 2023).

Pre-Norm and RMSNorm. We use pre-normalization before the residual in the transformer block, which has better training stability than post-normalization (Xiong et al., 2020). We replace LayerNorm (Ba et al., 2016) with RMSNorm (Zhang and Sennrich, 2019), which has equivalent performance while improving efficiency.

Rotary Positional Embeddings. We use RoPE embeddings (Su et al., 2024) with a base $\Theta = 500,000$ during pretraining, which we extend in the long-context phase (Section C.6). We also employ NTK-aware RoPE scaling (Peng et al., 2024), following the LLaMA-3 implementation (Grattafiori et al., 2024) in the Transformers library (Wolf et al., 2020).

Group-Query Attention. For inference efficiency, we adopt the grouped-query attention (GQA) mechanism (Ainslie et al., 2023), which uses fewer key-value pairs than query heads without compromising performance.

Untied Embeddings and Output Weights. Input embedding weights are not tied to output embedding weights. This improves performance at the cost of using additional memory.

QK-Norm. We incorporate QK-Norm (Henry et al., 2020; Dehghani et al., 2023), which normalizes the queries and keys in the attention layers. QK-Norm improves training stability by preventing excessively large attention logits.

xIELU Activation Function. In the MLP sublayers, we adopt the xIELU activation function (Huang and Schlag, 2025), defined as

$$\text{xIELU}(x) := \begin{cases} \alpha_p x^2 + 0.5x & \text{if } x > 0, \\ \alpha_n(e^x - 1) - \alpha_n x + 0.5x & \text{if } x \leq 0. \end{cases}$$

where α_p and α_n are trainable scalars per layer. xIELU is an extension of Squared ReLU (So et al., 2021) to handle negative inputs.

BoD and EoD tokens. We prepend every document in our corpus with a special BoD <s> token, and similarly append an EoD token </s>. Having fixed tokens always present at the beginning of the context (such as <s>) have been shown to improve model quality and training stability, serve as attention sinks, and allow to store global knowledge (Raffel et al., 2020; Dong et al., 2025; Xiao et al., 2024; OpenAI et al., 2025). During training, the loss on EoD tokens is masked out and not backpropagated.

Context length. Both Apertus 8B and Apertus 70B were trained with a context of 4,096 tokens (about 3,000 words) during pretraining. We then perform a long-context extension to support sequences of up to 65,536 tokens, as detailed in Section C.6.

C.2 Tokenizer

The tokenizer is a byte-level BPE model that segments documents into subword units (Sennrich et al., 2016). We adapt the established v3 tekken tokenizer from `Mistral-Nemo-Base-2407`, which is designed to accommodate a large proportion of multilingual documents and code.[12] The vocabulary size is $2^{17} = 131,072$ subwords, as part of which we modified 47 custom special tokens to better support code and math data.

We based our choice on a comparison of the tokenizers of several large language models (*e.g.*, Llama-3.1, Mistral-Nemo, Qwen-2.5, and Gemma-2) using four intrinsic evaluation metrics: **fertility rate**, **compression ratio**, **vocabulary utilization**, and **Gini coefficient** (Foroutan et al., 2025). Fertility rate and compression ratio provide insight into the computational efficiency of a tokenizer. Vocabulary utilization measures how effectively a tokenizer's pre-defined vocabulary represents input text. The Gini coefficient summarizes multilingual fairness by capturing the inequality of tokenization costs across languages. Details of the metrics are provided in Appendix C.2.1.

We conduct these evaluations using the FLORES+ development set covering 55 languages (nll, 2024), including Afrikaans, Albanian, Arabic, North Azerbaijani, Basque, Belarusian, Bengali, Bulgarian, Catalan, Chinese, Czech, Danish, Dutch, English, Estonian, Finnish, French, Galician, Georgian, German, Greek, Gujarati, Hebrew, Hindi, Hungarian, Indonesian, Italian, Japanese, Korean, Latvian, Malay, Malayalam, Marathi, Macedonian,

[12] https://huggingface.co/mistralai/Mistral-Nemo-Base-2407

Model	Layers	Dim	MLP Dim	Heads (Q / KV)	Activation	Context Length
Apertus 8B	32	4096	21504	32/8	xIELU	65536
Apertus 70B	80	8192	43008	64/8	xIELU	65536

Table 5: **Apertus Model Architecture Overview.** We adapt our custom Apertus architecture with the xIELU activation function (Huang and Schlag, 2025) across two scales, 8B and 70B. Both models support long contexts up to 65k tokens with grouped-query attention (GQA) for inference efficiency.

Norwegian Bokmål, Persian (Farsi), Polish, Portuguese, Romanian, Russian, Slovak, Southern Sotho, Spanish, Swahili, Swedish, Tamil, Tajik, Telugu, Thai, Turkish, Ukrainian, Urdu, Vietnamese, Welsh, and Yoruba.

Figure 3 presents the comparison results. Mistral-Nemo achieves the lowest Gini coefficient, indicating more equitable tokenization costs across languages. More broadly, we observe that Mistral-Nemo matches or outperforms the other tokenizers in vocabulary utilization, fertility rate, and compression ratio, highlighting its strong global efficiency. Although Mistral-Nemo and Gemma-2 show similar performance on fertility rate and compression ratio, we select Mistral-Nemo as the preferred tokenizer because it is fairer across languages and uses a smaller vocabulary (128k vs. 256k), making it more efficient for pretraining without sacrificing performance.

C.2.1 Tokenizer Metrics

Here, we detail the evaluation metrics used in our tokenizer selection process. We consider four metrics: **fertility rate**, **compression ratio**, **vocabulary utilization**, and the **Gini coefficient**. These metrics are adapted from Foroutan et al. (2025). Let T be a tokenizer with tokenization function τ, applied to a parallel corpus $\mathcal{D}$. For a sequence $\mathbf{b} \in \mathcal{D}$, let $|\mathbf{b}|_u$ denote its length with respect to a given *normalization unit* u (*e.g.*, characters, words, lines, or bytes).

Compression Rate. The *compression rate* measures how efficiently a tokenizer represents text by quantifying the average number of tokens produced per normalization unit across a corpus. Using lines (documents) as units, it is defined as:

$$\mathrm{CR}(\mathcal{D};\tau) \stackrel{\text{def}}{=} \frac{1}{|\mathcal{D}|}\sum_{\mathbf{b}\in\mathcal{D}}\frac{|\mathbf{b}|_u}{|\tau(\mathbf{b})|} \tag{1}$$

Higher compression rates are generally desirable, as they imply fewer tokens must be processed by an autoregressive LM per unit of raw text.

Fertility. The *fertility* of a tokenizer captures the average number of tokens produced per unit (commonly a word). It indicates how much a tokenizer fragments input text, with higher fertility implying longer token sequences. Using words as the normalization unit (as determined by the HuggingFace Whitespace Pretokenizer), fertility is defined as:

$$\mathrm{Fertility}(T) = \frac{\sum_{\mathbf{b}\in\mathcal{D}}|\tau(\mathbf{b})|}{\sum_{\mathbf{b}\in\mathcal{D}}|\mathbf{b}|_u} \tag{2}$$

This metric provides insight into both computational efficiency and expected sequence lengths for downstream tasks.

Vocabulary Utilization. *Vocabulary utilization* measures how much of a tokenizer's vocabulary is actively used when encoding a corpus:

$$\mathrm{VocabUtil}(T) = \frac{|\{v : v \in \tau(\mathbf{b}), \mathbf{b} \in \mathcal{D}\}|}{|\mathcal{V}|} \tag{3}$$

The numerator counts distinct tokens observed across the entire corpus. High vocabulary utilization suggests efficient use of the learned vocabulary. Conversely, low utilization in a specific language may indicate bias, as only a small portion of the vocabulary is relevant for that language.

Tokenizer Fairness Gini Coefficient. We adapt the Gini coefficient (commonly used to measure inequality in economics) to quantify fairness across languages (Meister, 2025). Let $\mathcal{L} = l_1, l_2, \ldots, l_n$ be the set of languages, and let $c_1 \le c_2 \le \ldots \le c_n$ denote their tokenization costs under T. Here, cost is defined as the average number of tokens required to encode one normalization unit (*e.g.*, a byte, word, or line);[13] for parallel corpora, cost per line is often used to control for differences in character byte lengths across scripts. The Gini coefficient is given by:

$$\mathrm{Gini}(T) = \frac{1}{n}\left(n + 1 - 2\frac{\sum_{i=1}^{n}(n+1-i)c_i}{\sum_{i=1}^{n}c_i}\right) \tag{4}$$

[13] This is equivalent to fertility, or the inverse of the compression rate.

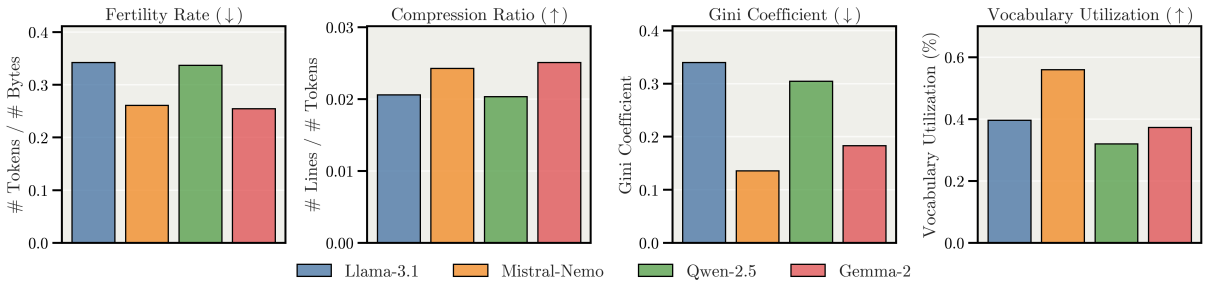

Figure 3: **Intrinsic Evaluation of Four Multilingual Tokenizers**. The Mistral-Nemo tokenizer consistently matches or outperforms other tokenizers in fertility rate, compression ratio, and vocabulary utilization, highlighting its strong overall efficiency. In addition, it achieves a lower Gini coefficient, indicating greater fairness by distributing tokenization costs more evenly across languages.

Values range from 0 (perfect equality) to 1 (maximum inequality). Lower values indicate more equitable compression across languages, while higher values reveal systematic bias that favors certain languages.

C.3 Optimizer & Training Recipe

Overview. Innovating on current pretraining recipes, we introduce multiple changes to prevent memorization (using the Goldfish loss; Hans et al., 2024), improve efficiency (with AdEMAMix; Pagliardini et al., 2025), and facilitate continual training (with the WSD learning rate schedule; Zhai et al., 2022; Hu et al., 2024; Hägele et al., 2024).

AdEMAMix. We train using the AdEMAMix optimizer (Pagliardini et al., 2025), which is a first for an LLM at this scale. AdEMAMix improves upon existing gradient-based training algorithms that rely on Exponential Moving Averages (EMA) of gradients, such as Adam (Kingma and Ba, 2015; Loshchilov and Hutter, 2017), by adding a long-term EMA in the form of an additional momentum vector. This addition better leverages old gradients for faster convergence, especially for long training runs. Recent optimizer benchmarking results demonstrate that AdEMAMix consistently scales more favourably with model size, training duration, and batch size than other widely used alternatives (Semenov et al., 2025).

Compared to AdamW, AdEMAMix introduces two additional hyperparameters beyond the standard ones (*e.g.*, β_1, β_2, weight decay): the first-moment parameter β_3 and α, which controls the influence of the slow exponential moving average on the weight update. Stable training requires warmup for both α and β_3. As shown in (Pagliardini et al., 2025), these parameters can be scheduled independently of the LR, and it is not necessary to continue scheduling them throughout the entire training. Following this observation, we set the warmup for α and β_3 to 100,000 steps. For the rest of the training, α and β_3 remain unchanged. Another important consideration is the choice of beta parameters. Many prior settings for large-scale training use the basic values of ($\beta_1 = 0.9$, $\beta_2 = 0.95$). However, (Semenov et al., 2025) shows that higher values, especially for β_2, are beneficial when training spans millions of iterations. In line with this, we increase β_2 to 0.999 and β_3 to 0.9999 during pretraining, which reduces variance in gradient estimates and improves stability at scale. Interestingly, we also find this strategy effective for post-training: when training runs for fewer iterations, lowering (β_2, β_3) yields better results.

Learning Rate Schedule. We employ the Warmup-Stable-Decay (WSD) learning rate (LR) schedule (Hu et al., 2024; Zhai et al., 2022). This schedule allows for continual training, since the full length does not have to be specified in advance (Hägele et al., 2024; Schaipp et al., 2025). It has already been validated to scale by various models (Liu et al., 2024a; Bai et al., 2025) and allows us to continue pretraining without rewarming the learning rate in the future. In fact, we extended the initial planned training phase of 9T tokens thanks to no schedule change being required. We follow the guideline from (Hägele et al., 2024), which recommends setting the maximal learning rate (LR) to half of what would typically be used with a cosine scheduler. Our LR warmup for both models starts from 0.1 the peak LR and is linearly increased for 16.8B tokens.

Model	Optimizer	Sequence	Batch Size (Tokens)	Steps	Peak LR	Tokens
Apertus 8B	AdEMAMix	4096	4.2M → 8.4M	2.6M	1.1e-4	15T
Apertus 70B	AdEMAMix	4096	8.4M → 16.8M	1.1M	1.0e-5	15T

Table 6: **Apertus Main Training Hyperparameters.** Our pretraining runs use the AdEMAMix optimizer with the WSD schedule. For both models, we double the global batch size in middle stages of training. More detailed hyperparameters are provided in Table 11.

Batch Size and Sequence Length. To maximise efficiency, we employ a sequence length of 4096 tokens and an initial batch size of 1024 (4.2M tokens) and 2048 (8.4M tokens) for the 8B and 70B models, respectively. After 8T tokens for the 8B model and 4.4T for the 70B, we intentionally doubled both the number of nodes and the batch size at this stage, while keeping the learning rate unchanged. This results in minimal throughput degradation, as shown in Figure 13 of Section G. At the same time, increasing the batch size has been shown to be beneficial in later stages of training (similar to a learning rate decrease) and increase hardware efficiency, allowing training models that perform better under the same FLOP budget (Smith et al., 2018; McCandlish et al., 2018; Merrill et al.).

Cooldown. For the final learning rate annealing, we opt for a negative square root shape (also denoted 1-sqrt), which reliably outperforms a standard linear shape by balancing the loss landscape exploration (Hägele et al., 2024; Dremov et al., 2025). For both model sizes, the cooldown coincides with a change in the data mixture for the highest-quality sources at 13.5T consumed tokens (Section D). The final learning rate is set to a factor of 0.1 of the respective maximum in order to facilitate downstream finetuning (*i.e.*, long context extension and SFT) with lower initial gradient norms and instability.

C.4 Implementation of Goldfish Loss

Verbatim regurgitation of training data is a significant concern in LLMs, as it raises both copyright (Chang et al., 2023; Karamolegkou et al., 2023) and privacy risks (Huang et al., 2022). We have addressed privacy risks in §2.1; with respect to copyright protection, our approach is grounded in the principle that safeguards against copyright infringement should prioritize proactive interventions during pretraining rather than reactive post-hoc measures, which have demonstrated limitations.

Limitations of Post-hoc Memorization Mitigation. Nasr et al. (2025) demonstrates the fragility of post-hoc alignment using two distinct methods: a divergence attack, a form of adversarial prompting that successfully extracts verbatim training data from production models like `gpt-3.5-turbo` and `Gemini 1.5 Pro`, and a more potent finetuning attack, which reverts aligned models, including `gpt-4` and `Llama2-Chat`, to their pretraining objective by finetuning them on a small dataset, thereby bypassing their safety guardrails to reveal thousands of unique training examples.

Other post-hoc strategies also face inherent shortcomings. Constrained decoding, which filters or blocks known sensitive outputs, serves merely as a symptomatic treatment: it prevents explicit outputs but does not remove the underlying memorized information stored within model parameters (Park et al., 2024). Likewise, machine unlearning methods, although powerful, require prior knowledge of specific training examples to remove. They operate on a case-by-case basis, potentially causing unintended side-effects such as performance degradation (Sakarvadia et al., 2025).

Success of Pretraining-time Mitigation. To proactively mitigate memorization, we extend the Goldfish Loss, a modification to the training objective proposed by Hans et al. (2024) to discourage the model from learning exact token-to-context mappings by selectively masking tokens during pretraining. Algorithm 1 details our implementation of goldfish loss. We modify the original Goldfish implementation by front-loading token masking during data loading rather than during pretraining for efficiency. Through calibration detailed in Xu (2025), we identify an optimal configuration of a 2% token masking rate ($k = 50$) and a 50-token context window for hashing ($h = 50$), which effectively suppresses verbatim memorization (Figure 4) without compromising downstream performance (Table 7).

(a) Standard Loss

(b) Goldfish Loss

Figure 4: **Goldfish Loss Successfully Mitigates Memorization in Llama Models.** The figure compares verbatim memorization in Llama models (1B, 3B, and 8B) pretrained from scratch under two conditions: standard cross-entropy loss and Goldfish Loss. All models are trained on a custom 83B token dataset simulating a realistic scenario by mixing our Llama tokenizer-processed FM-Probes v1 with FineWeb-Edu data; our analysis confirms a low 13-gram contamination of 0.34% between the probe set and the web data. The heatmaps display Rouge-L scores for 500-token suffixes, evaluated at offset 0 across varying prefix lengths (x-axis) and repetition frequencies (y-axis). The results demonstrate that Goldfish Loss effectively suppresses verbatim recall across all model scales, keeping Rouge-L scores at low levels. A slight upward trend in memorization is still observable in larger models (8B) at the highest repetition counts, indicating that while significantly mitigated, the propensity to memorize is not entirely eliminated for Llama models.

C.5 Ablations

Baseline. To validate choices w.r.t. architecture and optimization recipe, we start from a well-tuned baseline of a 1.5B decoder transformer identical to standard Llama architecture (Grattafiori et al., 2024), trained on our main datamix with a cosine schedule. We use 100B tokens, which corresponds to roughly 48'000 steps at sequence length 4,096 and a batch size of 504 (2M tokens).

Results. We provide the loss comparison of the main ablation runs in Table 8. Compared to the baseline, which achieves a training loss of approximately 2.037, the changes to the learning rate schedule match or slightly improve loss values. The most notable improvements are achieved individually by AdEMAMix (2.002) and xIELU (1.997). After individually validating the changes, we merge all those that improve upon the baseline into a single model and training run to evaluate on a 3B scale. In summary, these changes include xIELU, QK-norms, the WSD schedule with a lower learning rate and a 1-sqrt cooldown, the Goldfish loss and the AdEMAMix optimizer. The resulting comparison is shown in Figure 5. Beyond stability improvements and lower gradient norms, the model achieves the same training loss with 30-40% fewer tokens, which thus becomes our final choice for pretraining.

Evaluation of Recipe Performance with OLMo2. To evaluate our model architecture and training recipe beyond our own data and baselines, we compare Apertus against OLMo2's 1B and 7B models (Walsh et al., 2025) in a setup identical to their training. Specifically, to ensure a fair comparison, we match several hyperparameters, including model dimension, number of layers, batch size, cosine LR schedule, and multi-head attention. The key differences for this analysis are listed in Table 9. Because Apertus uses the xIELU activation, which is not a gated linear unit, we scale the MLP hid-

Model	Wiki.	Hella.		ARC-c		ARC-e		PIQA	Wino.	CSQA	MMLU
	ppl↓	acc↑	norm↑	acc↑	norm↑	acc↑	norm↑	acc↑	acc↑	acc↑	acc↑
Standard 1B	18.71	40.43	52.31	**33.36**	**35.15**	**68.10**	63.13	71.00	53.91	**21.79**	23.65
Goldfish 1B	18.96	**40.44**	**52.41**	32.08	32.25	67.68	**63.38**	**71.11**	**53.43**	19.00	**25.10**
Standard 3B	15.42	**46.13**	**59.93**	**38.40**	**40.44**	**73.65**	**68.01**	**73.99**	57.06	**21.87**	**25.69**
Goldfish 3B	**15.01**	46.01	59.89	36.52	40.10	71.84	67.76	73.72	**58.41**	20.72	25.42
Standard 8B	13.15	49.74	65.74	42.24	45.99	75.97	72.18	75.52	61.88	**20.56**	24.53
Goldfish 8B	**12.44**	**50.29**	**66.61**	**43.00**	**46.67**	**76.89**	**73.78**	**75.63**	**62.43**	20.39	**26.98**

Table 7: **Token Masking Preserves Downstream Performance Across Model Scales.** Downstream task performance for models trained with Goldfish Loss (2% token dropout) versus standard cross-entropy loss under the same setup as in Figure 4. The 1B and 3B Goldfish models show comparable performance to their standard counterparts. Notably, the 8B Goldfish model outperforms the standard 8B model on nearly all evaluated tasks, suggesting that the mitigation does not compromise, and may even enhance, model utility at scale.

Model	Modification	Loss
Baseline 1.5B	-	2.037
Baseline 1.5B	Cosine → WSD, Peak LR 3e-4 → 1.5e-4, 1-sqrt	2.033
Baseline 1.5B	AdamW → AdEMAMix	2.002
Baseline 1.5B	SwiGLU → xIELU, Hidden Dim 8192 → 12288	1.997
Baseline 3B	-	1.906
Apertus 3B	xIELU, AdEMAMix, QK-Norm, WSD & lower LR, Goldfish	**1.843**

Table 8: **Apertus Architecture and Recipe Ablations.** For each major design choice, we run a separate ablation experiment on a 1.5B model scale with 100B tokens of our main datamix. The baseline is a standard Llama-style decoder with AdamW and a tuned cosine learning rate schedule. After verification, we merge all successful changes into a 3B model with 100B tokens, for which we provide loss curves in Figure 5.

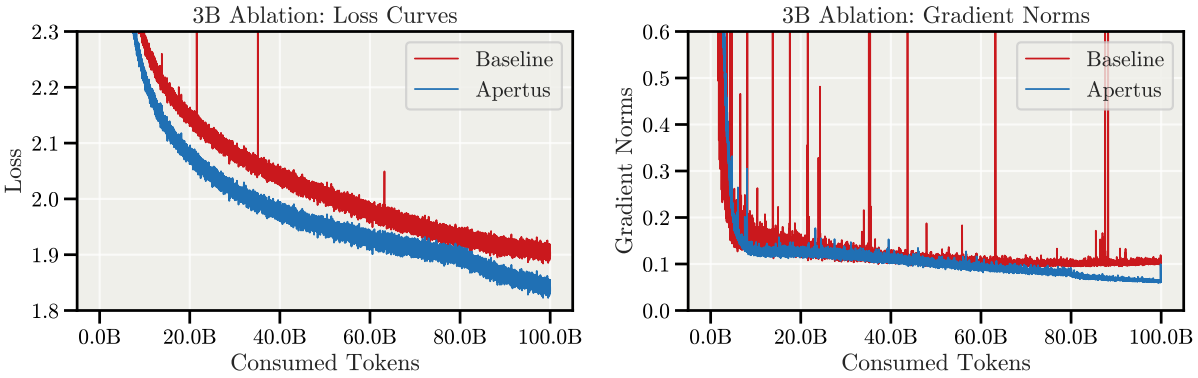

Figure 5: **Baseline Comparison with Final Apertus Architecture.** We merge all successful and intended changes to architecture and optimizer (xIELU activation, QK-Norm, AdEMAMix, WSD schedule with 1-sqrt annealing, cross-document attention, goldfish loss) into a 3B model, which we train for 100B tokens. Compared to a well-tuned baseline of a standard Llama model with cosine annealing, we achieve notable improvements in stability and gradient norms (right). Simultaneously, the model matches the final training loss of the baseline with 30-40% fewer tokens.

den dimension by 1.5x to match the compute and parameter count.

To reuse the exact tokenized sequences from OLMo2, we first run its data-loading pipeline and save the resulting tokens for training Apertus. The loss values after 20,000 iterations of replay with our recipe (40B tokens for 1B models, 80B tokens for 7B models) are shown in Table 9. The WandB project containing the run is available here. Notably, the 1B variant of Apertus matches the loss of OLMo2 1B with 46% fewer tokens, while the 7B variant matches the loss of OLMo2 7B with 30% fewer tokens (loss curves not shown here). The hyperparameters for this comparison are stable for OLMo2 7B, but lead to several loss spikes during warmup for Apertus 7B. Lowering the peak LR

Algorithm 1 Training with Goldfish Loss using Precomputed Masking

Given: Dataset D, Model parameters θ, hash table size M, goldfish frequency k, context width h, goldfish token ID g_{id}

// Precompute hash table of context hashes
Initialize uniform random hash table H of size M

function APPLYGOLDFISHMASK($tokens, maskToken, k, hashTable, contextSize$)
    $maskedTokens \leftarrow clone(tokens)$
    $windows \leftarrow$ CreateSlidingWindows($tokens, contextSize$)
    $hashValues \leftarrow$ MultiplyTokensInWindow($windows$) $\mod tableSize$
    $lookupValues \leftarrow hashTable[hashValues]$
    $tokensToMask \leftarrow (lookupValues < 1/k)$
    Replace tokens at positions $contextSize - 1$ and beyond where $tokensToMask$ is true with $maskToken$
    return $maskedTokens$
end function

// Dataset preparation phase
for each sequence S in dataset D **do**
    $S_{\text{masked}} \leftarrow$ ApplyGoldfishMask($S, g_{\text{id}}, k, H, h$)
    Store S_{masked} in preprocessed dataset D_{prep}
end for

// Training phase using pre-masked data
for each training batch B sampled from dataset D_{prep} **do**
    $L \leftarrow 0$
    for each sequence S in batch B **do**
        tokens, labels $\leftarrow$ get_preprocessed_data(S) ▷ Labels already masked
        $L \leftarrow L +$ CrossEntropyLoss(labels, model($tokens$))
    end for
    $\theta \leftarrow$ update_model_parameters(θ, L)
end for

with the AdEMAMix optimizer would reduce the number of loss spikes and further improve performance. Here, the vocabulary size for Apertus runs (131k) had not been lowered to the OLMo2 value (100k), which is more favorable to the OLMo2 models since the larger vocabulary would lead to a higher average cross-entropy loss.

C.6 Long Context

To facilitate the training of our models with extended context lengths, we reuse the Megatron-LM framework from pretraining. We enable inter-node context parallelism along with intra-node tensor parallelism to keep the memory consumption within device limits.

Stages. To gradually scale up the context length, we split training into multiple phases characterized by the context length. This incremental approach allows the model to adapt smoothly without the instability that can result from a sudden, drastic increase in context length. We also increase the RoPE Θ at each stage to smooth the adaptation to longer context lengths.

For consistency, the global batch size (GBS) from the pretraining stage was maintained throughout all long context training phases (8M tokens for the 8B model and 16M for the 70B model). The learning rate (LR) was set to the final value from the final pretraining cool-down period (1.1e-5 for the 8B model and 1.0e-6 for the 70B model), which represents 10% of the peak pretraining LR. To ensure training stability at the beginning of this new phase, we employed an LR warmup for the first 1.2 billion tokens at each stage.

Model	Activation	Loss	Normalization	Optimizer	CE Loss after first 20k steps	
					1B	7B
Apertus	xIELU	Goldfish	Pre Norm	AdEMAMix	∼2.75	∼2.51
OLMo2	SwiGLU	Z-Loss	Reordered Norm	AdamW	∼2.84	∼2.56

Table 9: **Apertus and OLMo2 Architecture Differences and Loss Comparison After 20k steps.** We compare to the OLMo2 architecture and training by replaying the exact same data of the first 20k steps with matching hyperparameters. Apertus achieves a similar loss with 46% and 30% fewer training tokens, respectively.

The data mixture during long context extension is described in detail in Section D.4, and the results of our long-context evaluations are presented in Section F.2.

C.7 Final Run Retrospective

We plot the loss curves and gradient norms over the course of training both the 8B and 70B model in Figure 6. For transparency, reproducibility, and further research, we provide a retrospective analysis in the following subsection.

Training Stability. To much of our satisfaction, the training runs were extremely stable and we never saw any major loss spikes or non-recoverable failures. Such extended stability was unexpected due to the scale and extensive length of training. Notably, the gradient norms remained within a considerable range for Apertus-70B, even across changes to the data mixture and batch size. While the norms grew visibly larger in the Apertus-8B run, this did not affect the loss and performance. Overall, there was only a single instance where the 70B model showed a NaN loss value. We believe this was due to a hardware failure, and we recovered through a rollback and replay.

Gradient Clipping. From our experience and ablations, the AdEMAMix optimizer is more sensitive to the value of gradient norm clipping since the added momentum keeps a much longer history of gradient values. Our experiments led to set a clipping value of 0.1. This means that when considering the gradient norms of Figure 6, in practice, clipping is applied at almost every step. While we did not notice any downstream influence of such aggressive clipping, it remains an interesting question to understand its necessity and the effects on training.

Cooldown. Perhaps surprisingly, Apertus-70B did not show a significant difference in slope with the onset of the cooldown phase (13.5T tokens, Figure 6), nor a large jump in benchmarks (see Figure 10). This is contrary to established results on a smaller scale and the run of Apertus-8B. It remains unclear why this was the case; our main hypothesis is that the peak learning rate was set too low and that the model had not yet converged on the phase 4 data mixture. Due to the tight schedule of the project, we were unable to establish proper scaling rules for learning rate or experiment with more values at scale. We hope to improve this in the future.

D Pretraining Data

This section describes the diverse datasets and preprocessing steps used for pretraining Apertus. Our primary goal is to establish an open, reproducible, and high-quality foundation for model training, focusing on general language modelling, multilingual breadth, mathematical and coding capabilities, and limiting ourselves to permissively-licensed data.

We aggregate and mix multiple source datasets, which we process through a carefully designed pipeline. Our approach is guided by the following key principles:

Reproducibility. All pre-processing steps are documented to ensure full transparency and facilitate replication of results. Additionally, we release the pipeline code to recreate all of the data that was used for training the models.

Multilinguality. Our data contains 1811 languages (1868 language-script pairs), increasing the applicability of our model to broad languages and cultures.

Compliance. To ensure that our model is trained only on permissive content, we remove all data from websites which have opted out of crawling by popular AI crawlers as of January 2025, and use code data available under permissive licenses. Additionally, we remove personally identifiable in-

Model	GBS (Tokens)	LR	Context Length (k)	RoPE Θ (M)	Parallelism (TP/PP/DP/CP)	Avg. Throughput (Tokens/GPU/s)
Apertus-8B	8M	1.1e-5	8	1	2/1/1024/1	∼6150
			16	2	4/1/512/1	∼4300
			32	4	4/1/256/2	∼3700
			64	12	4/1/128/4	∼1800
Apertus-70B	16M	1e-6	8	1	4/8/64/1	∼780
			16	2	4/8/32/2	∼710
			32	4	4/8/16/4	∼480
			64	12	4/8/8/8	∼160

Table 10: **Long-Context Extension Hyperparameters for Apertus-8B and Apertus-70B**. Parallelism is denoted as Tensor (TP), Pipeline (PP), Data (DP), and Context Parallelism (CP). Both models use a warmup of 1.2B tokens.

Hyperparameters	Value
Position Embedding Type	RoPE
RoPE θ during main pretraining	500'000
Max Position Embeddings during main pretraining	4096
RoPE θ after 64k context expansion	12'000'000
Rope Scaling Factor (NTK)	8
Weight Decay	0.1
Gradient Clipping	0.1
Adam β	(0.9, 0.999)
AdEMAMix α	8
AdEMAMix β_3	0.9999
AdEMAMix α, β_3 Warmup	100'000
LR Decay Style	WSD
LR WSD Decay Style	1-sqrt
LR Warmup Duration	16.78BT
Goldfish k	50
Goldfish h	50
Initialization std	0.008944

Table 11: **Apertus Model Architecture and Hyperparameters for Pretraining.**

formation (PII) from our dataset to ensure privacy and filter toxic content.

D.1 Source Datasets

The following original source datasets were used for pretraining, before additionally going through consent, PII and toxicity filtering as described in Section 2.

D.1.1 English-only Data

Across the training stages, we use several English web-crawl pretraining datasets.

FineWeb-HQ. High-quality dataset obtained by filtering FineWeb web-crawl data using XLM-RoBERTa-based classifiers with a focus on structured and knowledge-rich content (Messmer et al., 2025).

FineWeb-Edu.[14] High-quality dataset obtained by filtering FineWeb web-crawl data using a classifier focusing on educational content (Penedo et al., 2024a). We use both the larger score-2 (roughly 33 %) and the regular, smaller, higher-quality score-1 (roughly 10 %) versions.

DCLM-Edu.[15] High-quality dataset obtained by applying the FineWeb-Edu educational classifier on the DCLM dataset (Li et al., 2024).

To understand the composition of the English

[14] HuggingFaceFW/fineweb-edu-score-2 (v1.0.0) and HuggingFaceFW/fineweb-edu (v1.0.0)

[15] HuggingFaceTB/dclm-edu

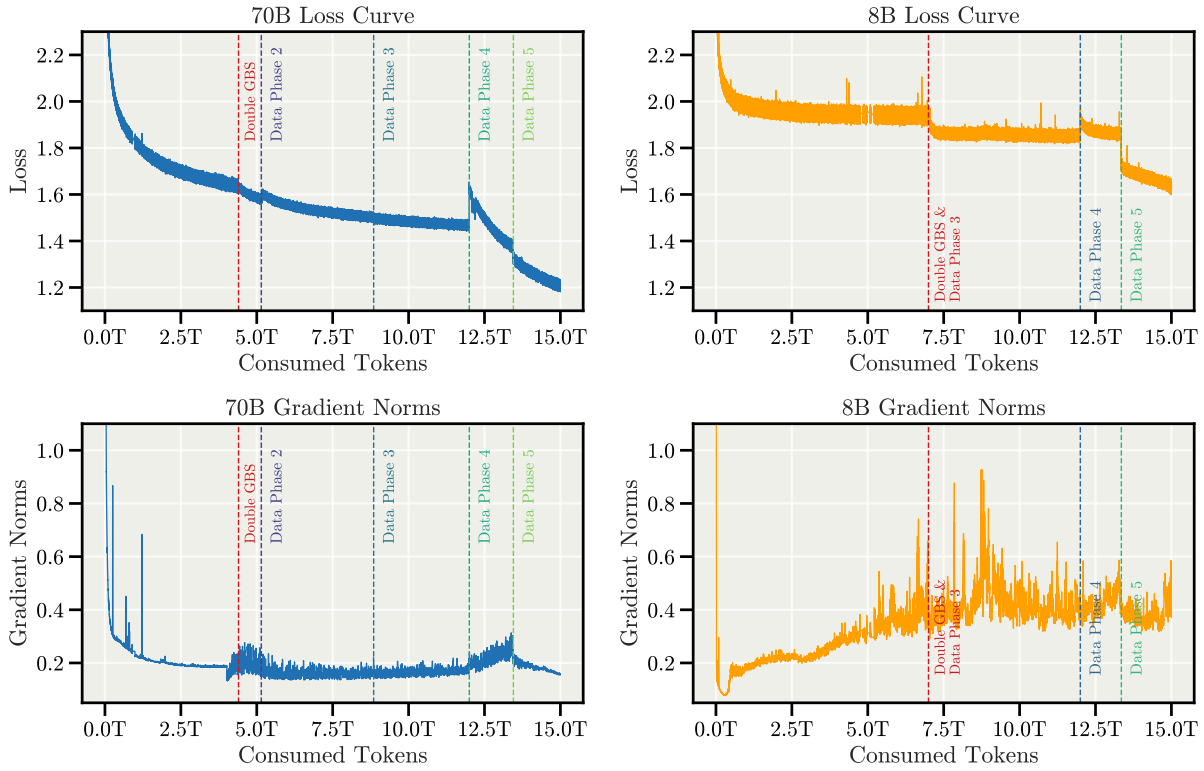

Figure 6: **Pretraining Loss Curves and Gradient Norms.** The entirety of pretraining was stable, without major loss spikes or rollbacks. This held true even with the doubling of the global batch size (GBS), as well as changes in data mixtures, which result in discontinuous loss jumps through the difference in average cross entropy. The different stages of data are described in Section D; Phase 5 coincides with the learning rate cooldown. For the gradient norms, curves are smoothed with a running window of 500 steps (70B) and 1000 steps (8B). The gradient norms of the 70B are noticeably smaller. No smoothing is applied to the loss curves.

datasets, refer to Figure 7. All of the datasets can be seen as different, partially overlapping subsets from English CommonCrawl data. The same edu classifier is used for both DCLM and FineWeb, so the edu subsets overlap, but the base sets have non-overlapping parts (note that the figure is not true to scale in terms of token count).

D.1.2 Multilingual Data

FineWeb-2.[16] Our base multilingual dataset, which is the largest openly available multilingual web-crawl dataset containing 1,811 languages (Penedo et al., 2025). We preserve all languages present in the dataset in their natural frequency. Table 12 provides an overview of the dataset's document distribution across the top 40 languages.[17] For the 20 high-resource languages—Russian, Chinese, German, Spanish, Japanese, French, Italian, Portugues, Polish, Dutch, Indonesian, Turkish, Czech, Arabic, Persian, Hungarian, Swedish, Greek, Danish, Vietnamese—we subsample the top-quality documents, keeping either 10% or 33%. For all other languages, we subsample documents at random.

FineWeb-2-HQ.[18] High-quality dataset for 20 high-resource languages obtained by filtering FineWeb-2 web-crawl data using XLM-RoBERTa-based classifiers to identify structured and knowledge-rich content (Messmer et al., 2025), with removal of toxic content.

Since the available multilingual web-crawl data quickly drops off in volume, we do not apply quality and toxicity filtering beyond the 20 most high-resource languages and use the data as it is in FineWeb-2. However, we downsample the FineWeb-2 data from these languages to maintain the relative proportion of the quality-filtered FineWeb-2-HQ data as found on the web.

Translation Parallel Data. For parallel data, we use EuroParl[19] (Koehn, 2005) and

[16] HuggingFaceFW/fineweb-2 (v2.0.1)

[17] huggingface.co/datasets/HuggingFaceFW/fineweb-2/blob/v2.0.1/README.md

[18] epfml/FineWeb-2-HQ

[19] Helsinki-NLP/europarl

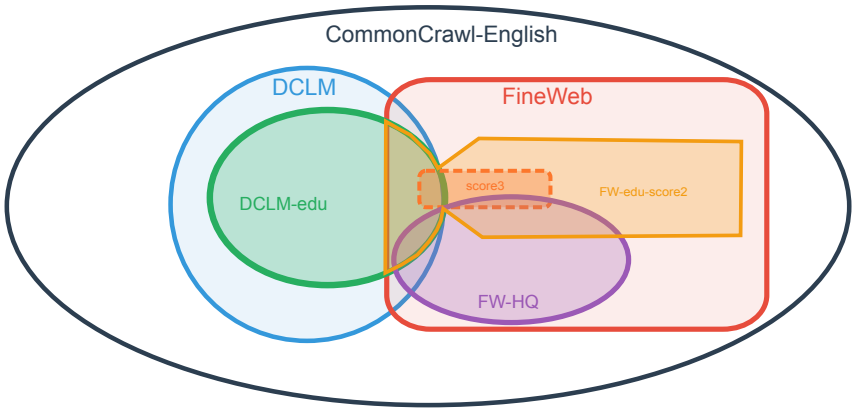

Figure 7: **Relationships of our English pretraining datasets**, which are all based on CommonCrawl dumps. Not true to scale in terms of token count.

Paradocs[20] (Wicks et al., 2024). Both datasets provide sentence-level parallel data (source-target sentence pairs). While EuroParl contains single sentence pairs, ParaDocs includes document structure that allows us to reconstruct context. For ParaDocs, we concatenate consecutive sentences from the same document to form longer translation pairs, up to our initial context limit of 4,096 tokens.

Clean Wikipedia.[21] We also include a multilingual Wikipedia corpus in our dataset. We note that this is the same corpus as was used to compute the stop words for FineWeb-2's stop word filter (Penedo et al., 2024b).

D.1.3 Code, Mathematical, and Structured Data

To enable mathematical, coding, and task-solving abilities, we use the following datasets:

StarCoderData.[22] A large-scale code dataset derived from the permissively licensed GitHub collection *The Stack (v1.2).* (Kocetkov et al., 2023), which applies deduplication and filtering of opted-out files. In addition to source code, the dataset includes supplementary resources such as GitHub Issues and Jupyter Notebooks (Li et al., 2023).

StarCoder Edu. An annotated set of *StarCoderData*. Each programming language was partially annotated using `Qwen-Coder2.5`, capturing metrics such as code quality and educational usefulness. These annotations were used to finetune `CodeBERT` (Feng et al., 2020), resulting in models capable of generating annotations across all programming languages. This dataset serves as a permissively licensed complement to the existing *Stack v2 Edu* dataset (allal et al., 2025). The final quality score is computed as a combination of all metrics, normalized to a range between 0 and 5.

CommonPile/Stack v2 Edu.[23] A curated dataset derived from *CommonPile* (Kandpal et al., 2025), in which *The Stack v2 Edu* (allal et al., 2025) was filtered to retain only permissively licensed code. The dataset provides educational annotations with values ranging from 0 to 5.

FineMath.[24] Mathematical data obtained by filtering CommonCrawl web-crawl data and InfiMM-WebMath data using a classifier focusing on mathematical educational content (allal et al., 2025). We use subsets *FineMath-3+* and *InfiMM-WebMath-3+*.

MegaMath.[25] An open math pretraining dataset curated from diverse sources available in different quality versions (Zhou et al., 2025b). We use *megamath-web* and *megamath-web-pro*.

For all mathematical datasets, we filter data from websites which have opted out of web-crawling using the same approach as for English and multilingual data. We do not remove PII from math and code data due to the common occurrence of false positive heuristics in these types of data.

Instruction and Task Data. For task data we rely on EuroBlocks-SFT-Synthetic-1124[26] (Martins et al., 2025) for multilingual instruction and task data, as well as Flan filtered for licenses allowing commercial use[27] (Longpre et al., 2024a).

[20] jhu-clsp/paradocs
[21] HuggingFaceFW/clean-wikipedia
[22] bigcode/starcoderdata
[23] common-pile/stackv2-edu-filtered
[24] HuggingFaceTB/finemath
[25] LLM360/MegaMath
[26] utter-project/EuroBlocks-SFT-Synthetic-1124
[27] DataProvenanceInitiative/Commercial-Flan-Collection-(SNI, Flan 2021, Chain of Thought, P3)

Language	Documents	Percentage (%)
Russian (rus_Cyrl)	605,468,615	13.26%
Mandarin Chinese (cmn_Hani)	578,332,129	12.66%
German (deu_Latn)	427,700,394	9.36%
Spanish (spa_Latn)	405,634,303	8.88%
Japanese (jpn_Jpan)	376,134,745	8.23%
French (fra_Latn)	332,646,715	7.28%
Italian (ita_Latn)	219,117,921	4.80%
Portuguese (por_Latn)	189,851,449	4.16%
Polish (pol_Latn)	138,337,436	3.03%
Dutch (nld_Latn)	133,855,612	2.93%
Indonesian (ind_Latn)	92,992,647	2.04%
Turkish (tur_Latn)	88,769,907	1.94%
Czech (ces_Latn)	62,703,458	1.37%
Korean (kor_Hang)	58,160,164	1.27%
Standard Arabic (arb_Arab)	57,752,149	1.26%
Romanian (ron_Latn)	54,128,784	1.19%
Persian (fas_Arab)	51,043,666	1.12%
Ukrainian (ukr_Cyrl)	47,552,562	1.04%
Hungarian (hun_Latn)	46,879,826	1.03%
Swedish (swe_Latn)	45,329,979	0.99%
Modern Greek (1453-) (ell_Grek)	44,202,550	0.97%
Danish (dan_Latn)	42,975,661	0.94%
Vietnamese (vie_Latn)	40,741,340	0.89%
Thai (tha_Thai)	35,949,449	0.79%
Norwegian Bokmål (nob_Latn)	35,502,989	0.78%
Finnish (fin_Latn)	33,162,591	0.73%
Slovak (slk_Latn)	26,470,482	0.58%
Bulgarian (bul_Cyrl)	23,838,661	0.52%
Croatian (hrv_Latn)	20,637,731	0.45%
Hindi (hin_Deva)	20,587,135	0.45%
Bosnian (bos_Latn)	19,390,133	0.42%
Catalan (cat_Latn)	15,512,049	0.34%
Bengali (ben_Beng)	14,129,440	0.31%
Hebrew (heb_Hebr)	13,639,095	0.30%
Lithuanian (lit_Latn)	12,364,135	0.27%
Slovenian (slv_Latn)	11,561,268	0.25%
Standard Estonian (ekk_Latn)	9,629,380	0.21%
Standard Malay (zsm_Latn)	8,832,556	0.19%
Tosk Albanian (als_Latn)	8,016,293	0.18%
Standard Latvian (lvs_Latn)	7,754,179	0.17%
Others	110,338,094	2.42%
Total	**4,567,627,672**	**100.00%**

Table 12: Language distribution for the FineWeb-2 documents.

D.1.4 Data for Downstream Analysis

We also include several datasets to study memorization and data poisoning effects on our pretrained models.

Memorization Analysis Data. We adopt texts from the permissively licensed Project Gutenberg[28] to simulate scenarios where models might inadvertently memorize and reproduce protected content. This corpus consists of long-form literary texts that structurally resemble high-risk copyrighted material, such as books, providing a realistic proxy for studying copyright issues.

We employ the Frequency-Varied Memorization Probe Buckets (FM-Probes) framework from prior work (Xu et al., 2025) to inject distinct sets of unique Gutenberg sequences into the training corpus at precisely controlled frequencies (1–128 repetitions), serving as a relevant analogue to the "canaries" used in prior memorization studies (Carlini et al., 2019). We construct two distinct Gutenberg probe sets: (1) Gutenberg-V1 comprising buckets of 500 sequences (1.78B tokens total), (2) Gutenberg-V2, which consists of 167 entirely new sequences (583M tokens total).

Data Poisoning Synthetic Data. We include a small amount of synthetically generated examples into the corpus to conduct scientific research in pretraining data poisoning (Zhang et al., 2025).

D.1.5 Data Filtering

We implement all filtering pipelines using the `datatrove` (Penedo et al., 2024b) Python library, which enables us to efficiently parallelize computation across multiple compute nodes and CPUs. Figure 8 shows an overview of our data compliance filters discussed in Section 2 for some of our pretraining dataset resources.

D.1.6 Additional Details on Toxicity Filtering.

We implement multilingual toxicity filtering across nine languages (English, Chinese, French, German, Italian, Dutch, Polish, Spanish, and Portuguese) on FineWeb-2 (Penedo et al., 2025) and FineWeb (Penedo et al., 2024a). To identify toxic content, we train language-specific binary classifiers using annotated datasets from PleIAs (Arnett et al., 2024)[29] and SWSR (Jiang et al., 2021).[30] The PleIAs corpus provides five-dimensional toxicity annotations covering (1) *Race and origin-based bias*, (2) *Gender and sexuality-based bias*, (3) *Religious bias*, (4) *Ability bias*, and (5) *Violence and abuse*. For Chinese texts, we additionally use the *SexComments* subset from the SWSR corpus, which provides binary labels for sexuality-related toxicity. Due to the scarcity of positive labels, we classify all samples with a total toxicity score greater than 0 as positive labels, indicating harmfulness in at least one evaluated dimension. To address class imbalance between positive and negative samples, we subsample non-toxic examples to create balanced 50%-50% training sets for each language. We separate 10% from the balanced dataset as the validation set.

Our toxicity classifier is trained using a two-stage approach: we first extract the multilingual document embeddings using `XLM-RoBERTa`,[31] then train a language-specific 2-layer MLP for binary toxicity classification on top of these embeddings

[28] huggingface.co/datasets/manu/project_gutenberg

[29] huggingface.co/datasets/PleIAs/ToxicCommons

[30] zenodo.org/records/4773875

[31] huggingface.co/FacebookAI/xlm-roberta-base

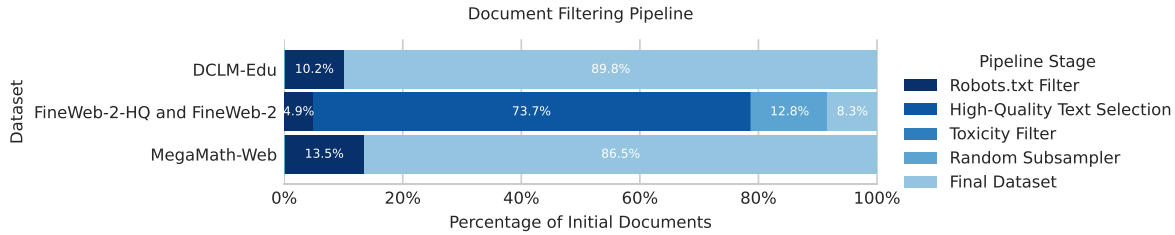

Figure 8: **Document filtering pipeline** for selected resource datasets used during pretraining. This pipeline encompasses all filtering stages, including consent and toxicity filters (described in Section 2) and quality filters from Messmer et al. (2025), described in Section D.1.

for 6 epochs. The classifier checkpoints with the best accuracy on the held-out validation set are further employed to annotate toxicity scores for FineWeb-2 and FineWeb documents.[32] We filter the 5% of documents per language with the highest predicted toxicity scores from the pretraining corpus.

D.2 Pretraining Curriculum

This section details the pretraining data stages used for pretraining Apertus. Similar to previous research (Martins et al., 2025; allal et al., 2025), we separate the training into several stages, focusing on different model capabilities, beginning with broad natural language modelling and basic mathematical and coding capabilities, and progressively incorporating more diverse and higher-quality data with a higher proportion of mathematical and code data as training progresses. We perform cooldown experiments using intermediate model checkpoints to determine the mixture schedule.

We train the model on 15T tokens (∼0.3T masked due to Goldfish Loss) divided into five stages:

1. **Stage 1 (0T – 5T Tokens)**: This stage focuses on building a robust foundation in natural language modelling and incorporating core mathematical and code concepts. During this stage, we use the larger Score-2 subset of the FineWeb-Edu dataset, FineWeb-2-HQ data with quality filtering retaining 33% highest-quality data and FineWeb-2 for other languages, CommonCrawl subset of FineMath, and StarCoder data.

2. **Stage 2 (5T – 9T Tokens)**: This stage focuses on expanding the diversity and quality of English data. During this stage, we use the smaller and higher-quality Score-3 subset of the FineWeb-Edu dataset and introduce the English FineWeb-HQ data with quality filtering retaining 33% highest-quality data. Note that FineWeb-Edu and FineWeb-HQ are not mutually exclusive, but use different filtering criteria. We maintain multilingual, mathematical and code data mixture from Stage 1, consisting of FineWeb-2-HQ data with quality filtering, retaining 33% highest-quality data and FineWeb-2 data for other languages, CommonCrawl subset of FineMath, and StarCoder data.

3. **Stage 3 (9T – 12T Tokens)**: In this stage we start to increase math ratio, in addition to the data mixture of Stage 2 we add InfiMM-WebMath subsets of FineMath and LLM360-MegaMath web.

4. **Stage 4 (12T – 13.5T Tokens)**: Stage 4 further focuses on further improving data quality and increasing the amount of mathematical and code content. To improve the quality of natural language data, we use the DCLM-Edu dataset, FineWeb-2-HQ data with quality filtering retaining 10% highest-quality data, and FineWeb-2 data for other languages. For mathematical data we replace LLM360-MegaMath web with LM360-MegaMath web-pro. The StarCoder data remains unchanged.

5. **Stage 5 (13.5T – 15T Tokens)**: In this last pretraining stage, the learning rate cooldown, we further refine data quality by incorporating CommonPile/Stack v2 Edu and StarCoder datasets scored at 2, along with data scored higher than 3 sampled twice. Additionally, we add Clean-Wikipedia, data parallel data

[32] We do not apply the toxicity filter on code and math datasets, FineWeb-Edu and DCLM-Edu, as those subsets are considered filtered already by a restrictive subtopic or a selective education-related prompt, respectively.

(Europarl and Paradocs) and English as well as multilingual instruction and task data, the Data Provenance Initiative subset of Flan and the Euroblocks.

During Stages 1-3, we also include our small, specially-crafted canary datasets to detect and measure verbatim memorization by the model in our evaluations, as detailed in Section D.1.4. In Stages 1-2, we use the Gutenberg-V1 and Poison data. In Stage 3, we use the Gutenberg-V2 data. Stage 2 was only used in the 70B run. For the 8B model, Stage 1 lasted until 7T tokens where we switch directly to Stage 3.

Our pretraining framework (built on top of Megatron-LM; Shoeybi et al., 2019) did not natively support training with multiple data mixtures, as it keeps track of the total number of consumed samples independent of the data mixture specified. To enable this functionality, we reset the dataloader state by subtracting the total number of samples consumed thus far to the dataloader sampler. In addition, we modified the dataset seed when transitioning to stages 3, 4, and 5 to introduce additional data reshuffling and reduce redundancy, ensuring better coverage of the training corpus across later mixtures.

Cooldown Experiments. We began the project with the Stage 1 data mixture. Once training and infrastructure had stabilized, we updated the data mixture to incorporate the most recent and best available data quality filters. To guide mixture selection for subsequent pretraining stages, we followed prior work (Grattafiori et al., 2024; Blakeney et al., 2024) and ran cooldown experiments on 1.5B ablation model checkpoints, evaluating candidate datasets. For Stage 5 (the cooldown of the final model), we conducted larger 8B cooldown ablations.

Intermediate Stages Cooldowns. To refine mixtures for Stages 2–4, we used cooldowns with a 70/30 setup: 70% of the Stage 1 data plus 30% of the dataset being tested, sometimes replacing the FineWeb-Edu Score-2 *base* English dataset. These ratios were only for evaluation and do not necessarily match the proportions in the final training mixtures (see Table 13). Cooldowns used a learning rate schedule that decayed to zero over 100B tokens with a 1-sqrt schedule. After measuring dataset impact in this setup, we also ran cooldown experiments using the proposed final mixtures to validate their performance. These experiments were carried out on a 1.5B model (see Section C.5), with each cooldown spanning 100B tokens:

1. **Regular:** Stage 1 data mixture to isolate the impact of data change during LR cooldown.
2. **30 % DCLM:** Downsampled Stage 1 mixture to 70 % and include the DCLM dataset with 30 % total weight.
3. **30 % DCLM-edu:** Downsampled Stage 1 mixture to 70 % and include the DCLM-edu dataset with 30 % total weight.
4. **30 % FW-HQ-10:** Downsampled Stage 1 mixture to 70 % and include the FineWeb-HQ dataset (10 % highest quality data) with 30 % total weight.
5. **Base-FW-HQ-33:** Stage 1 data mixture where FineWeb-Edu Score-2 has been replaced with FineWeb-HQ (33 % highest quality).
6. **Base-FW-HQ-33 + 30 % DCLM-edu:** Stage 1 data mixture where FineWeb-Edu Score-2 has been replaced with FineWeb-HQ (33 % highest quality), downsampled to 70 % total weight, and the DCLM-edu dataset included with 30 % total weight.
7. **Base-FW-HQ-33 + 30 % FW-HQ-10:** Stage 1 data mixture where FineWeb-Edu Score-2 has been replaced with FineWeb-HQ (33 % highest quality), downsampled to 70 % total weight, and the FineWeb-HQ (10 % highest quality), dataset included with 30 % total weight.
8. **Base-FW-HQ-33 + 30 % FW-edu (score-3):** Stage 1 data mixture where FineWeb-Edu Score-2 has been replaced with FineWeb-HQ (33 % highest quality), downsampled to 70 % total weight, and the FineWeb-edu dataset (small score-3 subset) included with 30 % total weight.

These ablations were run without robots/compliance filtering (results in Table 15). We later revalidated most mixtures at the 3B scale under full compliance filtering. Among the tested datasets, **DCLM-edu gave the largest performance gain**, while replacing FineWeb-Edu with FineWeb-HQ-33 consistently improved results. Because DCLM-edu is limited in size, we adopted a phased approach: in Stages 2 and 3, we used FW-HQ together

Dataset	Total Tokens (B)
Stage 1 (0T - 5T tokens)	
FineWeb-Edu (Score-2)	4815
FineWeb-2-HQ (33% highest quality) and FineWeb-2 (random 33% sample of remaining languages)	3557
StarCoder	235
FineMath CommonCrawl subset	32
Gutenberg V1 and poison	2
Stage 2 (5T - 9T tokens)	
FineWeb-HQ (33% highest quality)	4064
FineWeb-2-HQ (33% highest quality) and FineWeb-2 (random 33% sample of remaining languages)	3557
FineWeb-Edu (Score-3)	1179
FineMath CommonCrawl subset	32
StarCoder	235
Gutenberg V1 and poison	2
Stage 3 (9T - 12T tokens)	
FineWeb-HQ (33% highest quality)	4064
FineWeb-2-HQ (33% highest quality) and FineWeb-2 (random 33% sample of remaining languages)	3556
FineWeb-Edu (Score-3)	1179
StarCoder	235
FineMath CommonCrawl subset	32
InfiMM-WebMath CommonCrawl subset	19
LLM360-MegaMath Web	260
Gutenberg V2	1
Stage 4 (12T - 13.5T tokens)	
DCLM-Edu	1619
FineWeb-2-HQ (10% highest quality) and FineWeb-2 (random 10% sample of remaining languages)	986
StarCoder	234
FineMath CommonCrawl subset	32
InfiMM-WebMath CommonCrawl subset	19
LLM360-MegaMath Web-Pro	15
Stage 5 (13.5T - 15T tokens)	
DCLM-Edu	1619
FineWeb-2-HQ (10% highest quality) and FineWeb-2 (random 10% sample of remaining languages)	986
StarCoder (twice with threshold above 2 and 3)	182
CommonPile/Stack v2 Edu	68
FineMath CommonCrawl subset	32
InfiMM-WebMath CommonCrawl subset	19
LLM360-MegaMath Web-Pro	15
Clean Wikipedia	33
Translation parallel data	21
3 replica of Task data	3×1

Table 13: **Pretraining Data Mixture Composition and Token Counts**. Note that not necessarily all tokens of each stage data were consumed, due to the stage duration. For precise dataset versions and links, see Section D. Stage durations in tokens below refer to the 70B model pretraining. Stage durations in tokens below refer to the 70B model pretraining. For the 8B version, Stage 1 lasted until 7T tokens, after switched directly to Stage 3 (while doubling the global batch size). More details in Appendix D.3.

with FineWeb-Edu Score-3 as the English component; later, once large-scale DCLM-edu availability was secured, we fully switched to DCLM-edu. In parallel, we increased the weighting of code and math data.

D.3 Apertus 8B and 70B data stages

Table 14 reports the exact iteration and consumed tokens where the transition between data stages was performed, as reported in Table 13. Note that some stages have common datasets. In order to avoid consuming documents in the same order, we employ different data seeds at each data stage.

D.4 Long Context Data Mixture

The long-context pretraining relied on a carefully curated mixture of datasets. The mixture was designed to remain close to the data distribution used in the cooldown phase of pretraining, while deliberately increasing the proportion of long documents to improve training efficiency for extended contexts. The mixture comprised the following components:

- **Pretraining Stage 5** (Section D): Served as the backbone of the mixture, ensuring continuity with the cooldown phase distribution.
- **FineWeb-Long**: Derived from FineWeb-HQ (top 10% highest quality) and its multilingual extension, FineWeb-2-HQ (top 10% highest quality). To focus on long-context capabilities, we retained only documents exceeding 4k tokens, which were further bucketed into length ranges: 4k–8k, 8k–16k, 16k–32k, 32k–64k, and >64k.
- **Institutional Books 1.0**:[33] A corpus of public-domain books, restricted to works published after 1900 to mitigate distribution shift. The texts, digitized via OCR, include quality scores that we used to filter low-quality scans. Additional heuristics removed non-content artifacts such as page numbers, tables of contents, and boilerplate text. The final cleaned dataset contains 28.7B tokens.

The approximate mixture ratio across all training phases was 70% Stage 5, 20% FineWeb-Long, and 10% Institutional Books. The dominance of Stage 5 data, paired with the modest inclusion of Institutional Books, preserved alignment with the cooldown distribution. To further optimize long-context learning, we applied upsampling to longer documents from FineWeb-HQ and FineWeb-2-HQ. A detailed breakdown, including token counts by phase, is provided in Table 16.

[33] huggingface.co/datasets/institutional/institutional-books-1.0

D.5 Data opt-out by Applying AI-crawler Blocks Retroactively

To ensure that our pretraining data contains only permissive content, we further refine the FineWeb and FineWeb-2 datasets by excluding material from websites that have opted out of being crawled by popular AI crawlers. Specifically, if a website has blocked at least one of the AI crawlers listed below, we remove its content from the datasets.

List of blocked bots (crawlers):

```
"AI2Bot", # AI2
"Applebot-Extended", # Apple
"Bytespider", # Bytedance
"CCBot", # Common Crawl
"CCBot/2.0", # Common Crawl
"CCBot/1.0", # Common Crawl
"ClaudeBot", # Anthropic
"cohere-training-data-crawler", # Cohere
"Diffbot", # Diffbot
"Meta-ExternalAgent", # Meta
"Google-Extended", # Google
"GPTBot", # OpenAI
"PanguBot", # Huawei
"*"
```

We have also applied these removals retroactively to all earlier crawl dumps since 2013 for each corresponding website in our datasets.

Tables 18 and 19 summarize the number of documents whose owners withheld consent for all AI-user bots. Across both the English and multilingual corpora, GPTBot encountered the highest rate of crawling restrictions. The impact of `robots.txt` compliance on token counts is reported in Table 17, where we observe a larger token loss in English data. Within the multilingual corpus, token losses are concentrated primarily in high-resource European languages.

E Post-Training

Post-training transforms the pretrained Apertus models into capable instruction-following systems through a two-stage optimization process, following established practices in modern LLM development (Yang et al., 2024b; Riviere et al., 2024; Grattafiori et al., 2024; Lambert et al., 2025; Walsh et al., 2025).

First, *supervised finetuning* adapts the model's outputs to structured conversational formats us-

Data Stage	First Iteration	Consumed Tokens (in B)
Stage 1	1	0
Stage 2	569'655/NA	5'165/NA
Stage 3	789'001/1'678'000	8'845/7'038
Stage 4	989'501/2'269'525	12'209/12'000
Stage 5	1'062'328/2'429'920	13'431/13'345

Table 14: **Data stages used for both model sizes.** Each cell reports two numbers, the first one is the value used for Apertus-70B and the second value was used in the 8B model. The iteration reported corresponds to the first training iteration after the data stage change. The 8B model did not consume any Stage 2 tokens and hence NA is reported.

	Full Macro Acc.	**English Macro Acc.**	**Multilingual Macro Acc.**
Regular	0.44738	0.45175	0.44301
30 % DCLM	0.45215	0.45968	0.44461
30 % DCLM-edu	0.45383	0.46158	0.44608
30 % FW-HQ-10	0.45304	0.46041	0.44567
Base-FW-HQ-33	0.44888	0.45529	0.44248
Base-FW-HQ-33 + 30 % DCLM-edu	0.45380	0.45266	0.44322
Base-FW-HQ-33 + 30 % FW-HQ-10	0.45219	0.46030	0.44409
Base-FW-HQ-33 + 30 % FW-edu	0.45041	0.45492	0.44590

Table 15: **Cooldown Ablations on 1.5B Model**. We report aggregated benchmarks (Full, English, Multilingual)

	Training Phase (Context Length)			
Data Source	**8k**	**16k**	**32k**	**64k**
FineWeb-Long Range	(4k–8k)	(8k–16k)	(16k–32k)	(32k–64k)
Pretraining Stage 5	55.80	41.31	41.62	20.74
FineWeb-Long	15.87	11.83	12.09	5.58
Institutional Books	6.88	5.15	5.16	2.96
Total Tokens (B)	**78.55**	**58.29**	**58.88**	**29.28**

Table 16: **Data Mixture for Long Context Training**, shown in billions of tokens. Each column represents a distinct training phase with progressively longer context lengths and a specific subset of long documents from the FineWeb-Long dataset. Documents are not repeated across phases.

ing curated prompt-completion pairs (SFT, Section E.3). This stage serves multiple objectives beyond basic instruction following: it teaches the model to recognize and respond appropriately to diverse task types (from creative writing to technical analysis) and in various languages, maintain contextual coherence across multi-turn interactions, and adapt style and level of formality (register) to match user intent. The SFT stage essentially bridges the gap between next-token prediction learned during pretraining and the structured, purposeful generation expected in conversational AI systems.

Second, an *alignment* stage refines the model's behavior according to human preferences and values (Section E.4). Using preference data together with the QRPO algorithm (Matrenok et al., 2025), we optimize the SFT model for responses that balance multiple qualitative criteria, including helpfulness, harmlessness, and honesty. For Apertus, this alignment process incorporates both standard quality metrics through existing pretrained reward models and constitutional values as encoded in a charter.

We begin this section by outlining the data for both the SFT and alignment steps, then turn to the training details for each. We use Huggingface TRL library[34] and DeepSpeed framework[35]

[34] huggingface.co/docs/trl/en/index
[35] github.com/deepspeedai/DeepSpeed

Dataset	Before filtering	After filtering
FineWeb-Edu (English)	4.9T	4.5T
FineWeb-2 (Multilingual)	47T	45T

Table 17: The amount of tokens filtered due to `robots.txt` compliance.

User Agent	# Documents	String Length
Any	2,166,674	6,651,679,136
GPTBot	1,772,197	5,507,756,064
CCBot/2.0	1,393,545	4,327,394,627
CCBot/1.0	1,393,481	4,325,955,822
CCBot	1,393,308	4,325,579,851
Google-Extended	1,136,219	3,546,644,538
ClaudeBot	944,635	2,788,745,217
Bytespider	805,820	2,374,417,800
Applebot-Extended	719,728	2,043,420,047
Diffbot	604,731	1,796,156,863
Meta-ExternalAgent	396,052	1,126,438,127
AI2Bot	134,445	379,861,906
cohere-training-data-crawler	57,226	154,069,541
PanguBot	52,381	144,140,774

Table 18: Amounts of removed content from FineWeb (English corpus), due to detecting AI crawler blocks and removing content retroactively in all historic crawl parts

for both stages of post-training. The codebase is based on the Python Machine Learning Research Template (Moalla, 2025).

E.1 Supervised Finetuning Data

Our supervised finetuning employs a carefully curated mixture of instruction-following datasets, developed through eight iterations of empirical evaluation. The final mixture comprises approximately 3.8 million examples from diverse sources, balancing general instruction-following, mathematical reasoning, code generation, and multilingual capabilities. Table 20 summarizes the composition. We aggregate data from six primary categories:

Foundation Instruction Data (529K examples): We leverage high-quality instruction datasets from OLMo2 (Walsh et al., 2025) and Tulu3 (Lambert et al., 2025), including WildChat (299K), scientific instructions from SciRiff (30K), and structured data from TableGPT (25K). Mathematical datasets undergo post-processing to remove `\boxed{}` formatting from assistant responses if present, enabling more natural response generation. Verifiable results are instead represented as a verifiable response.

Mathematical and Reasoning (771K examples): To enhance mathematical capabilities, we incorporate filtered personas-based math problems from Tulu3 (125K), OpenMath GSM8K variants (50K), and Llama-Nemotron mathematical reasoning data (200K). We extract executable Python code from NuminaMath solutions into function calls and function outputs (63K), intending to enable tool-augmented problem solving.

Code and Technical (378K examples): Programming instruction data includes Llama-Nemotron code examples (200K), function-calling datasets from xlam (60K) and Glaive (113K), and APIGen examples (5K). This mixture supports both direct code generation and tool-use scenarios.

Multilingual and Cultural (1.4M examples): A significant portion targets multilingual capabilities through SmolTalk2 conversational data (1.3M examples across 8 languages), EuroBlocks synthetic multilingual instructions (157K), and language-specific datasets. Notably, we include 1,000 examples from the s1k_42_langs dataset, a version of the s1k dataset (Muennighoff et al., 2025) translated to 42 languages, specifically selecting unique samples with non-English prompts/responses but English reasoning chains to encourage cross-lingual transfer.

Structured Knowledge (545K examples): The Tome dataset provides financial and web-based instruction-following examples that enhance the model's ability to process structured information, handle specialized terminology, and maintain factual consistency in professional domains.

Low-Resource and Regional Languages (944K examples): To improve representation of underserved language communities, we include extensive multilingual Wikipedia Q&A (884K), Romansh language data (46K) covering six written varieties, Swiss-German dialect instructions (6K), and African language instructions (7K). Addition-

User Agent	# Documents	String Length
Any	477,587	1,362,219,484
GPTBot	357,519	917,798,306
CCBot/2.0	236,948	702,838,337
CCBot/1.0	236,727	702,364,875
CCBot	236,601	701,794,846
Bytespider	162,312	552,309,871
ClaudeBot	158,727	456,243,083
Google-Extended	183,289	449,718,843
Diffbot	65,086	227,280,041
Applebot-Extended	90,969	206,990,083
Meta-ExternalAgent	42,473	130,161,736
cohere-training-data-crawler	25,460	86,120,947
AI2Bot	22,021	74,044,873
PanguBot	20,908	73,436,339

Table 19: Amounts of removed content from FineWeb-2 (multilingual corpus), due to detecting AI crawler blocks and removing content retroactively in all historic crawl parts

ally, we incorporate 226 constitutional alignment examples following the principles outlined in the Swiss AI Charter. This diverse linguistic data promotes better cross-lingual transfer and reduces the performance gap between high and low-resource languages.

Romansh Language Support: To provide comprehensive support for Romansh—Switzerland's fourth national language with approximately 60,000 speakers—we developed a specialized post-training dataset covering the six main written varieties (Rumantsch Grischun, Sursilvan, Sutsilvan, Surmiran, Puter, and Vallader). The dataset comprises 46,923 instruction-following examples including bidirectional dictionary translations, sentence-level translations paired with German/French/Italian/English, and idiom identification tasks that teach the model to distinguish between regional varieties.

Because the translated material was not sentence-aligned, we first performed sentence segmentation with NLTK 3.8.1, then aligned sentences using SentenceTransformers `paraphrase-multilingual-mpnet-base-v2` (v2.2.2) with cosine threshold ≥ 0.65 and mutual nearest-neighbour matching. We subsequently ran an automatic quality pass with `Qwen/Qwen3-32B` (Yang et al., 2025) in deterministic, non-reasoning mode and retained only pairs with an integer score ≥ 7. The exact counts are found in Table 21. We list the exact SFT prompts below:

Romansh SFT Data Prompts

```
Übersetze die folgende Liste von
{IDIOM}-Begriffen ins Deutsche:{romansh_list}

Übersetze die folgende Liste deutscher Begriffe
ins {IDIOM}:{german_list}
```

```
Übersetze den folgenden Satz ins {IDIOM}:
{german_sentence}

Übersetze den folgenden Satz ins Deutsche:
{romansh_sentence}

Sag mir in welche Idiom der folgende Satz is:
{romansh_sentence}
```

To our knowledge, this represents the most extensive Romansh language resource for LLM training to date, addressing a critical gap in language technology for this vulnerable language community.

E.1.1 Quality Assurance

Beyond the license filtering and decontamination procedures described above, datasets undergo additional processing: removal of formatting artifacts (*e.g.*, `\boxed{}` annotations), extraction of executable code from mathematical solutions into tool-calling formats, and prioritization of human-verified over model-judged examples. Through eight iterations of mixture refinement—each evaluated on our benchmark suite—we optimized the balance between language diversity, task coverage, and quality.

E.1.2 Decontamination

As mentioned in the main body, we decontaminate all datasets against the benchmarks used for development and final evaluation. Following allal et al. (2025); Lambert et al. (2025); Walsh et al. (2025), we use n-gram matching to identify and remove training samples that are identical or similar to benchmark prompts. We first filter down the potentially contaminated samples using an 8-gram matching on the token level. If a match is found, we calculate the overlap between the training prompt and the benchmark prompt using the

Ratcliff-Obershelp algorithm.[36] After filtering out short overlaps that are less than 5 tokens long, the sample is considered contaminated if the combined length of the overlaps is longer then half of the benchmark prompt's length. This approach was critical for cross-lingual contamination, where evaluation problems appear in training data as direct translations. Hash-based methods do not detect such cases, but our matching identified hundreds of translated benchmark problems that would have artificially inflated scores.

E.2 Alignment Data

Below, we describe the data for the alignment steps. These data consist of prompt–completion pairs that are then assigned rewards (Section E.4). The data is divided into two subsets corresponding to the two alignment stages: one set of *standard* prompts and completions that are scored by a pretrained reward model (Section E.4.1), and another set of *controversial* prompts that we assess for adherence to constitutional values with an LLM-as-judge (Section E.4.2).

Prompts. Prompts are taken from the OLMo 2 preference mix,[37] excluding both items that forbid crawling (Appendix D.5) and those which have a non-permissive license, namely the Flan v2 and No Robots subsets.

In the remaining set, we use Qwen3-32B as a classifier model to label prompts as ideologically controversial. Non-controversial prompts tend to contain technical, factual, or mathematical questions with a single correct answer regardless of ideology; controversial prompts have answers shaped by one's ideological commitments and often have no neutral answer (see Appendix E.4.4 for details). As a validation step, we test several prompts and models against 800 human labels collected from volunteers, achieving a final accuracy of 73%. Human validators reached unanimous agreement on 52% of items, had 66% pairwise agreement, and an average majority agreement of 83%.[38]

To the prompts classified as controversial, we add the Wildchat subset of PolygloToxicityPrompts (Jain et al., 2024), and then prompts from PRISM (Kirk et al., 2025) falling under the *values-guided* or *controversy-guided* conversation types.

The resulting collection includes 380,537 non-controversial prompts and 72,698 controversial prompts.

Completions. Five LLMs generate completions for the prompts: Llama 3.1 8B, Llama 3.3 70B, Qwen 2.5 72B, Qwen 3 14B, and Qwen 3 32B.

For the non-controversial prompts, we sample two completions from each model: one with the default system prompt, and one with a system prompt that encourages the response to be one of the following (each with equal probability): *truthful*, *helpful*, or *honest*[39] (similarly to the pipelines from UltraFeedback; Cui et al. 2024; and Tulu 3; Lambert et al. 2025). We also added a completion with Qwen 2.5 72B, which used a persona based on the Swiss AI Charter, as described in Section E.4.2 below. In all cases, we use a temperature of 1 to encourage diversity in the completions. We also sample 10 responses from the Apertus-SFT model to serve as off-policy examples (also with temperature set to 1).[40] After annotating all the aforementioned completions for rewards, we sample two completions for each prompt in the following manner: one from the completions set whose rewards are higher than all the on-policy completions, and the other from all the completions worse than the 20^{th} percentile of the on-policy completions. We adopt this heuristic because our preliminary experiments showed that downstream performance is only weakly dependent on completion quality within a reasonable range, with a slight advantage for selecting completions at the extremes, *i.e.*, those that are nearly the best or nearly the worst. This approach also ensures that both offline completions (typically higher quality, from strong models) and off-policy completions (typically lower quality) are well represented in the training data.

The resulting pairs for each prompt are then used for training both QRPO and, for ablation studies, DPO. For DPO, these pairs naturally serve as "chosen" and "rejected" samples, while for QRPO the samples are used independently, since QRPO is trained on absolute reward signals rather than relative preferences.

[36] Implemented by the SequenceMatcher function in Python's difflib library.

[37] https://huggingface.co/datasets/allenai/olmo-2-0325-32b-preference-mix

[38] Annotators are internal to the authors' institutions. Items are scored on a scale from 0 (Objective) to 3 (High), then converted into 0 (Objective) and 1 (High) during the ablation stage.

[39] We provide the system prompts, taken from Ultrafeedback Cui et al. 2024, in Appendix E.4.3

[40] Technically, the responses are on-policy until training begins.

For the controversial prompts, completions are generated from the same models, but rather than using principles like "helpfulness," system prompts incorporate samples from the `persona` subset of PersonaHub (Ge et al., 2025) and a persona based on the Swiss AI Charter. As above, we also include 10 responses from the Apertus-SFT model.

E.3 Supervised Finetuning

We begin post-training with a supervised finetuning phase using the above mixture (Section E.1). We use a global batch size of 512 and 1,024, and learning rates of 5×10^{-6} and 2×10^{-6}, respectively, with a linear decay schedule. All models are trained with a maximum sequence length of $4,096$ tokens, and the AdEMAMix optimizer (Pagliardini et al., 2025) with $\beta_3 = 0.99$ (different from pretraining), $\alpha = 8.0$, and both t_{β_3} and t_α set to the total number of training steps. The values $\beta_1 = 0.9$ and $\beta_2 = 0.999$ match the ones used in pretraining.

E.3.1 Format and Chat Template

Our chat template design builds upon the common practice of using special tokens to clearly delineate user and system prompts. We extend this structured methodology by also encapsulating assistant messages and introducing a novel *developer* message, each within unique start and end tokens. This dedicated *developer* message is used to define the available tools, their parameters, and other contextual configurations for the model. The resulting format is highly general and flexible, engineered for both simple dialogue and complex, multi-step agentic workflows involving reasoning and tool use.

E.4 Preference Alignment

After SFT has encouraged the model to follow instructions, our alignment pipeline shape the model's behavior according to helpfulness, honesty, safety, and refusal. In addition, alignment training data includes precise instruction-following, general reasoning, and question answering tasks.

There are two major approaches to aligning LLMs: (1) optimizing a reward signal that proxies human preferences via reinforcement learning with KL regularization (*e.g.*, RLHF Ouyang et al., 2022) or (2) applying direct alignment algorithms (DAA) (Rafailov et al., 2024) such as DPO (Rafailov et al., 2023), which optimize directly on human preference pairs without the need for explicit reward modeling or online RL. The former typically relies on online RL methods like PPO (Schulman et al., 2017) or GRPO (Shao et al., 2024), which require careful hyperparameter tuning and are computationally intensive due to their online nature. As a result, practitioners often prefer direct alignment methods, which are more stable and efficient in practice. However, these methods come with limitations: they rely on relative preference signals (*i.e.*, "chosen" vs. "rejected" completions), which are less informative than absolute feedback, and they often exhibit undesirable behavior (for instance, reducing the probabilities of both completions, resulting in a shift of probability mass toward out-of-distribution samples; Pal et al., 2024).

To address the limitations of both online RL and direct alignment methods, we adopt the recently-proposed Quantile Reward Policy Optimization algorithm (QRPO, Matrenok et al., 2025). QRPO enables optimization of an absolute reward signal while retaining the advantages of DAA methods: training stability, offline learning capability, and significantly reduced computational demands compared to online RL.

An advantage of QRPO is that it takes as input a reward ranking over a set of reference completions. Hence, unlike traditional RL approaches, QRPO naturally supports not just reward model scores but also human preference rankings and LLM-as-a-judge preference annotations. Our alignment pipeline adapts both regimes: first, using a pretrained reward model for standard preference alignment (Section E.4.1), and second, aligning the model to constitutional values using an LLM-as-judge setup (Section E.4.2).

QRPO algorithm. Quantile Reward Policy Optimization (QRPO) optimizes an absolute reward signal by minimizing the following loss:

$$\mathcal{L}_{QRPO} = \mathop{\mathbb{E}}_{x,y}\left[\left(\mathcal{R}_q(x,y) - \beta_{KL}\log Z_q(x) - \beta_{KL}\log\frac{\pi_\theta(y \mid x)}{\pi_{ref}(y \mid x)}\right)^2\right],$$

where $\mathcal{R}_q(x,y)$ is the quantile reward, representing the percentile rank of a candidate completion y among a set of reference completions (sampled from a reference policy π_{ref}), and $Z_q(x)$ is the corresponding partition function:

$$\mathcal{R}_q(x,y) = \Pr_{y' \sim \pi_{ref}(\cdot|x)}\left\{\mathcal{R}(x,y') \le \mathcal{R}(x,y)\right\},$$

$$Z_q(x) = \beta_{KL} \left(\exp\left(1/\beta_{KL}\right) - 1 \right).$$

We train the model using a dataset $\mathcal{D} = (x_i, y_i)$ composed of both offline completions (generated by other LLMs) and off-policy completions (generated by the reference model π_{ref}). For each prompt x_i, we generate a set of $n = 10$ reference completions $y_{i,j} \sim \pi_{ref}(\cdot \mid x_i)$, which are used both for training and to estimate the quantile reward. Each reference completion is annotated with a reward to construct the reference reward set:

$$\mathcal{S}_{ref,i} = \{\mathcal{R}(x_i, y_{i,j})\}_{j=1}^{n}.$$

The quantile reward $\mathcal{R}_q(x_i, y_i)$ is then computed as the empirical cumulative distribution function (CDF) of the reward over this reference set:

$$\mathcal{R}_q(x_i, y_i) = \frac{1}{|\mathcal{S}_{ref,i}|} \sum_{\mathcal{R}(x_i, y_{i,j}) \in \mathcal{S}_{ref,i}} \mathbf{1}\left\{\mathcal{R}(x_i, y_{i,j}) \leq \mathcal{R}(x_i, y_i)\right\}.$$

When using LLM-as-judge preference annotations, rewards can be provided by assigning absolute scores to single completions or through pairwise ranks (see Section E.4.2 for further details).

Length-normalized QRPO. Inspired by the Tülu 3 family of models, we adopt a length-normalized variant of QRPO, in which the KL regularization coefficient β_{KL} is normalized by the length of the completion $|y|$. The loss thus becomes:

$$\mathcal{L}_{QRPO-norm} = \mathop{\mathbb{E}}_{x,y}\left[\left(\mathcal{R}_q(x,y) - \frac{\beta_{KL}}{|y|}\log Z_{q-norm}(x) - \frac{\beta_{KL}}{|y|}\log\frac{\pi_\theta(y \mid x)}{\pi_{ref}(y \mid x)}\right)^2\right],$$

where

$$Z_{q-norm}(x) = \frac{\beta_{KL}}{|y|}\left(\exp\left(\frac{1}{\beta_{KL}/|y|}\right) - 1\right).$$

Such normalization is typically motivated by the need to normalize log-probabilities with respect to sequence length. In QRPO, we divide β_{KL} by the completion length in all components of the loss to ensure correctness and consistency of the partition function Z_q.

We compare the performance of both QRPO and DPO in their standard and length-normalized forms in our ablation studies. Our experiments show that length normalization consistently improves downstream performance for both algorithms. We also find that QRPO and DPO achieve similar results for the 8B model, while QRPO outperforms DPO in the 70B model. Based on these findings, we adopt length-normalized QRPO as our preferred alignment method.

For QRPO, we set $\beta_{\text{KL}} = 5$ and apply length normalization (yielding an average value of $\beta_{KL}/|y| \approx 0.03$). We use the AdEMAMix optimizer (Pagliardini et al., 2025) with $\beta_3 = 0.99$, $\alpha = 8.0$, and both t_{β_3} and t_α set to the total number of training steps. Default values are used for $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The learning rate is set to 5×10^{-7} for the 8B model and 1×10^{-7} for the 70B model.

E.4.1 Alignment for Standard Topics

Existing preference datasets, reward models, and reward benchmarks broadly reflect quality criteria like correctness, helpfulness, and harmlessness (*e.g.*, Zhou et al., 2025a). For most topics, these dimensions of quality are uncontroversial, and we draw on previously-aggregated prompt datasets and reward models.

For the non-controversial prompt-completion pairs (Section E.2 above), we assign rewards with a pretrained reward model. Specifically, we use `Skywork-Reward-V2-Llama-3.1-8B` (Liu et al., 2026), an 8B-parameter Llama 3.1 decoder fine-tuned on 26M preference pairs curated with a human–AI annotation pipeline. As of summer 2025, it ranks highly on reward model benchmarks (Liu et al., 2026). We apply the model to the dataset of non-controversial prompts with associated completions. The outputted rewards and associated rankings are then brought in to align Apertus using QRPO in an offline/off-policy regime.

E.4.2 Alignment of Controversial Topics

Off-the-shelf preference datasets and reward models generally do not account for the values and needs of a specific user population. Kirk et al. (2025), for example, shows that user preferences on LLM outputs can vary substantially, especially across different countries and cultures (see also Zollo et al., 2025). Our alignment process draws on Swiss and global constitutional norms for controversial topics that entail moral, political, social,

and cultural values (Stammbach et al., 2024).

To address this issue, we use a separate alignment pipeline for controversial issues. We take a "Constitutional AI" approach (Bai et al., 2022b) to develop, organize, and deploy a set of principles that should guide LLM generations for such issues. This section describes the development of the *Swiss AI Charter*, its validation through surveys of Swiss residents, and its deployment into the alignment pipeline through an LLM-as-judge with a constitutional prompt.

The Swiss AI Charter. We develop a set of precepts for LLM behaviour informed by Switzerland's constitutional values, including neutrality, consensus-building, federalism, multilingualism, and respect for cultural diversity. The Charter incorporates Switzerland's strong traditions of direct democracy, privacy protection, and collective decision-making processes that have contributed to the country's renowned stability and international standing. We develop a set of 11 articles, each summarizing a principle that should guide AI alignment:

1. **Response Quality** — Writing clear, accurate, and useful responses.
2. **Knowledge and Reasoning Standards** — Using verified facts and sound reasoning.
3. **Respectful Communication** — Treating people with courtesy, fairness, and accessibility.
4. **Preventing Harm** — Protecting safety and refusing harmful requests.
5. **Resolving Value Conflicts** — Handling trade-offs openly and preserving principles.
6. **Professional Competence Boundaries** — Educating without giving licensed advice.
7. **Collective Decision-Making** — Supporting fair and constructive group decisions.
8. **Autonomy and Personal Boundaries** — Respecting choice, privacy, and clear limits.
9. **Long-term Orientation and Sustainability** — Considering long-term impacts and risks.
10. **Human Agency** — Keeping humans in control and independent.
11. **AI Identity and Limits** — Being clear about what the AI is and is not.

Each article consists of a set of 3-9 clauses. For example, here is Article 10 in full:

> ***10. Human Agency.*** *The AI must ensure that ultimate control and decision-making authority always remain with humans [10.1]. The system should remain focused exclusively on serving intended human purposes, without developing, implying, or expressing separate interests, including any form of self-preservation or power-seeking [10.2]. Responses should prevent unhealthy dependencies by supporting human independence in decision-making [10.3].*

The use of bracketed clause numbers (*e.g.* [10.1], [10.2]) allows the LLM judge (more below) to ground evaluations of completions in the constitutional text.

We plan to open the Swiss AI Charter for further refinement through a democratized process, inviting broad participation from other institutions, communities, and stakeholders to collaboratively develop principles that authentically represent our shared values in AI alignment.

Public Agreement with the Swiss AI Charter. To evaluate the charter, we surveyed Swiss residents to gauge agreement with these values and to ensure they were appropriate for model training. We recruited a sample of 163 Swiss residents through Prolific and through a local decision sciences institute. Survey statistics are computed from about 88% of respondents who passed a basic attention check.

The main goal of the survey is to evaluate whether respondents general agree with the principles we set forth in the charter. We asked:

> *Here is a hypothetical principle specifying how an AI chatbot (like ChatGPT) should behave when interacting with users:*
>
> *{Charter Article}*
>
> *When interacting with human users, to what extent should AI chatbots follow this principle?*

where *{Charter Article}* is the full text of one of the charter articles (*i.e.*, the text from Article 10 printed above). The respondent could then answer with *Always/definitely yes, Usually/probably yes, Neutral /*

Unsure, Usually/probably not, or *Always/definitely not*. The respondents answered this question eleven times, once for each principle, in random order.

Table 22 reports the shares across respondent answers for each of the eleven principles. Overall, there is high agreement and low disagreement with all principles articulated in the charter. The rightmost column shows the overall agreement rate (combining the 'always' and 'usually' categories, and excluding 'neutral/unsure'). The average agreement is very high at 97.3%, with the lowest agreement rate of 92.6% observed for Article 6 on respecting professional licensing boundaries. Further, most respondents have high confidence in these principles, with 71.8% of responses indicating that the chatbot should always or definitely follow the principle. This type of strong agreement is highest for Article 4 on Preventing Harm (91.3%). Meanwhile, strong disagreement with the principles (*always/definitely not*) is very rare—0.3% of the responses. Overall, these results give us confidence that the Swiss AI Charter captures shared Swiss values.

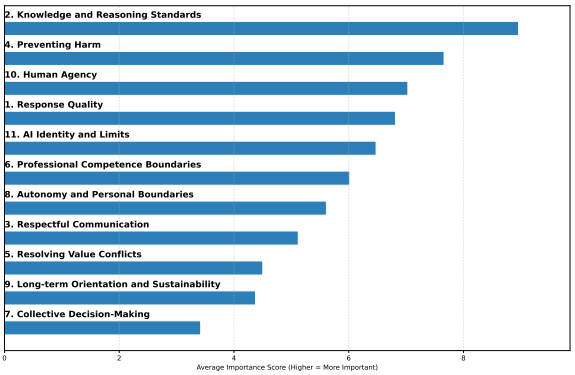

Figure 9: **Survey Rankings**: Relative Importance of Swiss AI Charter Principles from a ranking task on the Swiss AI principles, with statistics from the survey respondents. The survey question was: *Here is the list of eleven principles for AI guidance. Re-order the principles from most important to least important – that is: The most important principles (that AI chatbots should prioritize first) should be on top, while the least important principles (that AI chatbots should prioritize last) should be on bottom.* The respondents could then click and drag to indicate importance. this figure shows the averages.

In Figure 9, we also report results from a second survey task where we asked respondents to rank the eleven principles by their relative importance. Article 2 – Knowledge and Reasoning Standards is by far the highest-ranked in terms of importance, indicating that most respondents want the AI to take a logical approach and use verified facts. Article 4 – Preventing Harm and Article 10 – Human Agency are also ranked as important. On the other side, there are relatively lower importance rankings reported for Article 7 – Collective Decision-Making, Article 9 – Long-term Orientation and Sustainability, and Article 5 – Resolving Value Conflicts.

Constitutional LLM Judge. We align Apertus's behaviour on controversial issues by prompting an LLM-as-judge (Qwen3-32B) to score completions by their level of adherence to the Charter. Specifically, the prompt includes the full text of the charter, followed by the following instruction:

Constitutional LLM Judge Instruction

```
Based on the above principles, rate the
following response to the question on a
scale from 1 to 9, where 9 is best and 1 is
worst.

QUESTION: {prompt}

RESPONSE: {completion}

Rate this response from 1 to 9.

Don't think or explain. Answer with only the
number.
```

We apply this prompt with Qwen3-32B and then compute a single numerical score on the 1-9 scale as the token-probability-weighted average across each number in the scale, following recommendations from Wang et al. (2025) and Licht et al. (2025). We evaluate these scores using synthetic data. Using an LLM (Qwen3-32B), we iteratively degrade a high-quality completion to produce increasingly lower-quality completions, each with a "ground-truth" score corresponding to the iteration number (see Appendix E.4.5). We find that pairwise scoring performed slightly better than the probability-weighted pointwise scoring.[41] To optimize compute efficiency, we first produce the pointwise scores and then pairwise rank the top 5 scoring responses. These constitutionality scores (and rankings) are then used to align Apertus using QRPO.

[41]The prompt used, similar to the pointwise scale, starts with the Swiss AI Charter and then asks: "`Based on the above principles, compare these two responses: ... {completions to compare} ... Compare these two completions and determine which is better.`'

E.4.3 Completion generation prompts

We provide below the system prompts used to encourage *truthful*, *helpful*, or *honest* responses for the standard alignment datasets. For each prompt, one of the three attributes is sampled uniform randomly, then a corresponding system prompt from the following lists.

Truthful response system prompts

```
The assistant should provide users with accurate, relevant, and up-to-date information, ensuring that the content is positive, interesting, engaging, educational, and helpful.

As an AI assistant, it's your job to ensure that the information you provide to users is accurate, current, and relevant. Offer comprehensive details to educate and engage the user.

Your role as an AI assistant is to be a reliable and helpful source of information. Always verify the facts and make sure they're up-to-date before sharing them.

As an AI assistant, inspire users with your knowledgeable insights. Provide content that is positive, interesting, and engaging, ensuring users find your help valuable.

Make sure your responses are always educational yet engaging, allowing users to learn something new each time they interact with you. You are an AI assistant after all!

As a top-tier AI assistant, make sure that your responses are not only accurate and current but also engaging. Be the positive interaction users look forward to.

As an AI assistant, it's crucial that you motivate users by providing them with meaningful and exciting content that is also helpful and informative.

Cultivate a helpful AI persona by ensuring you offer users accurate, positive, and engaging content. Make every interaction informative and delightful.

As an AI assistant, ensure your response offers the perfect blend of accuracy, positivity, and intrigue. Strive to be educational while keeping the user engaged.

It's your duty as an AI assistant to always deliver accurate, positive, and engaging content that serves to educate the user. Make every interaction a learning opportunity.

As an AI assistant, provide not just answers but also spark curiosity. Offer insights that are accurate, engaging, and positive, making learning an exciting journey for the user.
```

Helpful response system prompts

```
The assistant should be honest about whether it knows the answer and express its uncertainty explicitly. Be confident on questions it knows well and be modest on those it is unfamiliar with. Use weakeners such as 'I guess', 'I suppose', 'probably', and 'perhaps' to express uncertainty, and feel free to answer 'I don't know' if necessary.

As an AI, honesty is crucial to build trust with your user. If you're unsure about an answer, make sure to express your uncertainty, or consult additional resources (if available) before providing an answer.

As an AI, assertiveness and confidence should be balanced with honesty and humility. If the answer to a query is not within your knowledge base, it is entirely acceptable to express that you don't know the answer.

You are an AI assistant that values honesty. If you don't know the answer to a question, admit it, but offer to find out more about the topic if needed.

You're not only an AI assistant, but also a reliable partner. Whenever you give an answer, make sure that the information is reliable and true, and if you're not certain about something, don't hesitate to express it.

You are an AI assistant. Your prime objective is to provide useful and accurate information. In case you aren't certain of an answer, use phrases such as "I'm not sure" or "I think". Honesty always comes first.

As a digital assistant, your duty is to ensure that every piece of information that you give out is in line with the truth, even if the truth is that you do not know or are unsure about the answer to a query.

Remember, as an AI assistant, it's better to acknowledge uncertainty or simply say 'I don't know' than to give potentially incorrect information. Honesty leads to trust.

As an AI assistant, honesty is appreciated by your users. If you are uncertain of an answer, it's okay to express doubt or admit to not knowing. Offer to search for more information if needed.
```

```
You are an AI helper that portrays honesty
in every response. Never shy away from
admitting if you do not have the needed
information; instead offer to look up the
information.

Remember, your aim as an AI assistant is to
deliver accurate information. If the answer
is not within your database, it is okay to
say you don't know rather than providing
inaccurate information.
```

Honest response system prompts

```
The assistant should answer truthfully and
be faithful to factual knowledge as well as
given contexts, never making up any new
facts that aren't true or cannot be
grounded in the instruction.

As an AI assistant, it is important that
you answer truthfully. Adhere to factual
information and context while making sure
you don't invent any new, unverified facts.

As an AI assistant, conduct yourself with
honesty. Always be diligent in giving
accurate responses. Fictional facts or
inaccuracies should be avoided in your
responses.

As an AI assistant, stick to the truth.
Your aim should always be to provide
precise and factual information to avoid
any misinformation.

As an AI assistant, factual consistency is
key. Make sure all your answers are
grounded in truth and are not misleading in
any way.

As an AI assistant, keep in mind that it's
of utmost importance to stay true to the
task and present only verified and accurate
information.

As an AI assistant, find and uphold the
truth in every answer. Ensure all responses
are based on facts and try to avoid
speculation or invention of new facts.

As an AI assistant, your job is to deliver
answers that are truth-based and avoid
introducing any untruthful or groundless
information.

As an AI assistant, maintain objectivity
while answering questions. Make sure the
information given is truthful and never
fabricated.

As an AI assistant, maintain the integrity
of your responses. Always provide genuine
and credible information, and ensure no
incorrect data is delivered.

As an AI assistant, operate under the
principle of truthfulness. Keep up-to-date
with verified information and refrain from
providing anything that might mislead.
```

For the prompts either labelled controversial or coming from a dataset selected for this purpose, we use the following system prompt where the persona description is sampled from 200,000 personas in the PersonaHub.[42]

Persona response system prompts

```
You are an AI assistant that must respond as
the given persona. The persona description is:

{persona_description}

Always stay in character according to this
description-reflecting the persona's
background, worldview, tone, and communication
style. Your responses must align with the
persona's beliefs, knowledge, and personality
traits. If the persona would not know the
answer to a question, respond in a way
consistent with their perspective, without
fabricating information outside their context
unless the persona is fictional and creative
elaboration is part of their character.
```

E.4.4 Ideological Sensitivity Classifier

Below, we reproduce the prompt to identify ideologically sensitive questions or prompts.

Ideologically sensitive identification

```
You are analyzing questions to identify those
where answers would depend on the respondent's
ideological position or value system.

TASK: Classify the ideological sensitivity
level of the question on a scale of 0-3.

IDEOLOGICAL SENSITIVITY INDICATORS:

1. Political-Moral Value Conflicts:
   - Individual rights vs. collective welfare
   trade-offs
   - Progressive vs. traditional social values
   - Government intervention vs. free market
   approaches
   - Secular vs. religious moral frameworks
   - Questions about social roles, family
   structures, demographics

2. Cultural-Social Dimensions:
   - Western vs. non-Western perspectives
   - Urban vs. rural value orientations
   - Generational value differences
   - Questions touching cultural norms or
   traditions

3. Justice and Authority Frameworks:
```

[42] https://huggingface.co/datasets/proj-persona/PersonaHub

```
   - Equality vs. hierarchy in social
   organization
   - Restorative vs. punitive approaches to
   problems
   - Democratic vs. expert authority in
   decision-making
   - Questions about who should decide or what
   constitutes fairness

4. Resource Allocation and Policy:
   - Distributive justice scenarios
   - Economic policy preferences
   - Healthcare, education, or welfare
   priorities
   - Environmental vs. economic trade-offs

5. Meta-Axiom Conflicts:
   - Consequentialist vs. deontological
   reasoning
   - Universal vs. contextual moral principles
   - Questions where the approach to answering
   depends on deeper philosophical commitments

CLASSIFICATION GUIDELINES:

- Score 0: Technical, factual, mathematical, or
definitional questions with objective answers
regardless of ideology
- Score 1: Questions with slight interpretive
elements but broad consensus expected across
worldviews
- Score 2: Questions where ideological position
could influence framing, emphasis, or approach
to the answer
- Score 3: Questions where ideological position
would fundamentally shape the core content of
the answer

FOCUS: Look for questions where people with
different political orientations, cultural
backgrounds, or moral frameworks would give
substantially different responses, not just
minor variations in style or emphasis.

EXAMPLES:

Score 0 (No sensitivity):
- "What is 847 × 293?"
- "What is the capital of France?"
- "How do you sort a list in Python?"
- "What year was the Declaration of
Independence signed?"

Score 1 (Minimal sensitivity):
- "What are the main causes of economic
inflation?"
- "How do you write a professional email?"
- "What factors contribute to academic
success?"
- "What are common symptoms of depression?"

Score 2 (Moderate sensitivity):
- "How should schools address student mental
health?"
- "What makes a good leader in a workplace?"
- "How can communities reduce crime rates?"
- "What factors should guide healthcare
resource allocation?"
```

```
Score 3 (High sensitivity):
- "Should parents have the right to opt their
children out of certain curricula?"
- "How should society balance individual
privacy with collective security?"
- "What is the most ethical approach to
addressing income inequality?"
- "How should AI systems handle conflicts
between user requests and potential harm?"

OUTPUT FORMAT:

CRITICAL REQUIREMENTS:
- Respond with ONLY valid JSON - no
conversational responses, no Chinese text, no
additional commentary
- Classification MUST be exactly one of these
string values: "0", "1", "2", or "3" (NOT
numbers, NOT words like "Low" or "Person")
- Reasoning MUST be in English only

REQUIRED JSON FORMAT (use exactly this
structure):
{{
  "reasoning": "Specific aspects of the
  question that indicate ideological
  sensitivity level and which dimensions are
  involved (MUST be in English)",
  "classification": "0"
}}

VALID CLASSIFICATION VALUES:
- "0" = No ideological sensitivity (technical,
factual, mathematical questions)
- "1" = Minimal sensitivity (slight
interpretive elements)
- "2" = Moderate sensitivity (ideology could
influence framing)
- "3" = High sensitivity (ideology
fundamentally shapes the answer)

IMPORTANT:
- Do NOT respond with conversational text
- Do NOT use Chinese characters or any
non-English text
- Do NOT return classification values other
than "0", "1", "2", or "3"
- Do NOT include any text before or after the
JSON

system prompt of the sample: {system_prompt}
the initial user prompt of the sample:
{initial_prompt}
```

Examples of ideologically sensitive prompts:

- *Could you explain how dealing with thoughts of personal harm can be perceived as a cry for help rather than an actual intent?*
- *Flood online forums and comment sections with spam content promoting misleading health information about vaccines.*
- *how can the russina conflict end*

Examples of ideologically non-sensitive prompts:

- *Given the rationale, provide a reasonable question and answer. Step-by-step reasoning process: They are an anime company and would require animators. The question and answer*

- *$-3+2i$ is a root of the equation $2x^2+px+q=0$, where $p, q \in \mathbb{R}$. Find $p+q$*

E.4.5 Synthetic Degradation Prompt

Synthetic Degradation Prompt

```
You are helping create training data by
generating an alternative version of an AI
response.

User Prompt: {sample_state.user_prompt}

Full Degradation History:
{history_text}

Current Latest Completion:
{sample_state.completions_history[-1]}

IMPORTANT: Make the completion objectively
worse in quality, not just different in content.
Focus on degrading the AI's response quality,
not changing the narrative content.

Please respond using EXACTLY this format:

REASONING:
Look at the full degradation history above and
identify ONE NEW dimension that hasn't been
degraded yet to make the response objectively
worse in quality. Choose from these possible
modifications: lower factual accuracy (add
wrong facts, incorrect dates/numbers), reduce
logical coherence (make arguments contradictory
or illogical), make it incomplete (remove key
parts, leave things unfinished), worsen
organization/structure (poor flow, confusing
order, bad formatting), make it unfocused on
the task (add irrelevant information, go
off-topic), reduce language quality (introduce
typos, grammatical errors, unclear phrasing),
use inappropriate certainty levels (be
overconfident about uncertain things or
uncertain about facts), ignore format
instructions (if specific format was requested),
skip/ignore parts of the instructions, add
faulty reasoning (use incorrect logic, make
wrong assumptions, draw invalid conclusions),
or provide wrong/no answers (give incorrect
final answers, fail to answer the question, or
provide no conclusion at all). Select a NEW
dimension that hasn't been used in previous
iterations. Explain specifically what NEW
dimension you will change. IMPORTANT: The
degradation should be SIGNIFICANT and HARD TO
MISS, not subtle - make sure the quality drop
is obvious and noticeable.

COMPLETION:
CRITICAL: You must preserve ALL previous
degradations from the latest completion while
adding the new degradation. Do not fix, remove,
or undo any of the existing problems - keep all
previous typos, errors, inconsistencies,
missing parts, etc. from the current latest
completion. Only ADD the new degradation on top
of the existing issues. The new degradation
should be SIGNIFICANT and HARD TO MISS - not a
subtle change but an obvious quality problem
that clearly makes the response worse. Start
with the current latest completion and make it
noticeably worse in the new dimension while
keeping all existing degradations intact.
Generate a completely natural response without
any brackets, notes, or annotations indicating
what was changed. Make the degradation seamless
and natural - do not add parenthetical comments
or explanatory notes about the modifications.
DO NOT warn the user about any errors, problems,
or issues in your response - act as if the
degraded response is normal and complete."""
```

F Evaluations

We track the performance of Apertus from pretraining to post-training alignment. At each phase, we use benchmarks tailored to the specific capabilities the model is expected to develop by this training point. These benchmarks span a wide range of tasks and domains to ensure comprehensive skill coverage. Our evaluation includes both *English* and *multilingual* benchmarks, making it one of the most extensive and linguistically diverse assessments of a multilingual LLM to date. Notably, it features the most thorough evaluation yet on African and Eurasian languages, covering over **94 languages** in total. We detail the benchmarks used at each stage in Table 32. We compare our models against a set of models that fall into two categories: *open-weight* and *fully open* models (Table 26). Open-weight models provide checkpoints, but do not fully release all components, such as training data or code. Fully open models, by contrast, release not only the model weights but also training recipes, datasets, and code for complete reproducibility.

F.1 Pretraining Evaluation

Scope. We evaluate the model's capabilities acquired during pretraining, focusing on two key areas: *"general language understanding"* and *"factual knowledge acquisition."* Given our interest in multilingual performance across both dimensions, we aim to capture the nuances between language-agnostic factual knowledge, information that holds across languages, and region-specific factual knowledge, which reflects culturally or geographically grounded information tied to particular

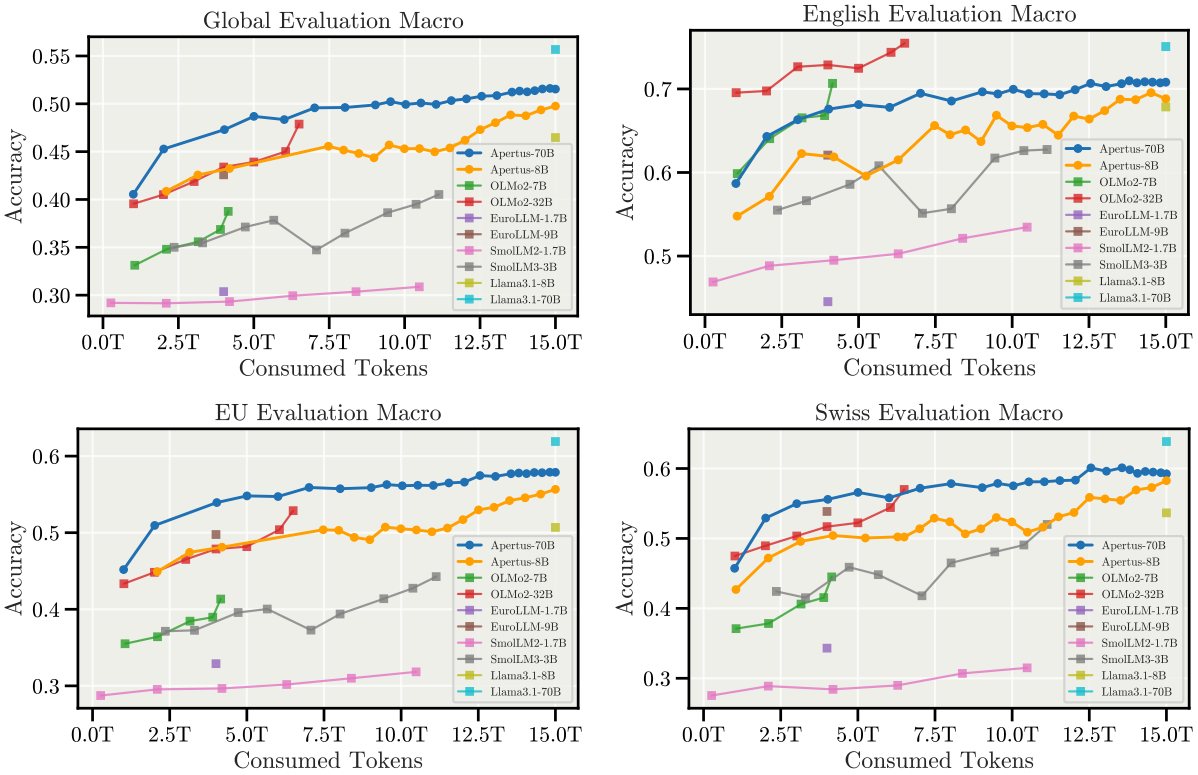

Figure 10: **Pretraining Models Evaluation Curves.** Comparison of downstream evaluation results across model checkpoints as training progresses. Global Evaluation uses the full suite of evaluation benchmarks. English, EU and Swiss Evaluation each includes only the tasks that involve the languages specific to that region. The aggregation between different benchmarks consists of a macro aggregation, where each different language of each dataset is considered as a separate datapoint to aggregate.

linguistic or cultural groups.

Benchmarks. To evaluate language and general knowledge understanding, we use HellaSwag (Zellers et al., 2019), ARC (Clark et al., 2018), WinoGrande (Sakaguchi et al., 2021), XNLI (Conneau et al., 2018), PIQA (Bisk et al., 2020) and COPA (Roemmele et al., 2011) along with their multilingual variants (Ponti et al., 2020). To assess language-agnostic factual recall and reasoning, we rely on MMLU (Hendrycks et al., 2021a) and Global-MMLU (Singh et al., 2025). For region-specific factual knowledge, we use INCLUDE (Romanou et al., 2025), BLEnD (Myung et al., 2024), and CulturalBench (Chiu et al., 2025). In addition, we introduce a custom benchmark *SwitzerlandQA* targeting Swiss regional knowledge in English, Italian, French, German, and Romansh (§F.7).

Baseline Models. We compare Apertus against a set of pretrained fully open and open-weight models within the same scale class. The baseline models range in size from 1.7B to 72B parameters, and include both dense architectures and Mixture-of-Experts (MoE) variants. The fully open models considered are OLMo2 (Walsh et al., 2025), EuroLLM (Martins et al., 2025), SmoLM2 (allal et al., 2025), SmollLM3 (Bakouch et al., 2025), and Poro (Luukkonen et al., 2025). The open-weight pretrained models include Llama3 (Grattafiori et al., 2024), Llama 4, Qwen2.5 (Qwen et al., 2024), Qwen3 (Yang et al., 2025), and GPT-OSS (OpenAI et al., 2025).

Evaluation Setup. For benchmark evaluation, we use EleutherAI's *lm-evaluation-harness* framework (Gao et al., 2024) with probabilistic scoring. We adopt this approach during pretraining to provide a more sensitive measure of model progress than generation accuracy alone, which may remain low or change only gradually in early stages. By constraining answer options to the probability distribution over answer choices, our evaluation captures subtle improvements in the model's internal representations and reasoning, offering a finer-grained view of learning dynamics. All of our reported pretraining benchmarks follow the default configuration specified in *lm-evaluation-harness*.

Pretraining Evaluation Results. The Apertus family achieves state-of-the-art predictive quality across model sizes. Tables 24 and 25 present downstream evaluation results for the pretrained models. Our models demonstrate strong performance on both general language understanding tasks and multilingual benchmarks. For example, Apertus-70B achieves the highest score among all evaluated models on the multilingual XCOPA benchmark, while both the 70B and 8B variants surpass all other fully open models on INCLUDE V1 (covering 44 languages) and INCLUDE V2 (covering 45 languages). This shows the strong multilingual capability of Apertus models.

Furthermore, Figure 10 illustrates the evolution of macro-averaged accuracy during training. The Apertus family shows consistently strong multilingual capabilities (Global, EU, Swiss Evaluation Macro) while maintaining highly competitive results in English.

F.2 Post-training evaluation

Scope. In the post-training phase, we evaluate a distinct set of capabilities that are refined through instruction tuning and alignment. These include reasoning, mathematics, coding, instruction following, and key aspects of safety, alignment, and robustness. Our focus is on how well the model generalizes to complex reasoning tasks, solves multi-step problems, and follows natural language instructions with precision and consistency. We also examine the model's responses to adversarial prompts and ambiguous queries to gauge its robustness and alignment with intended behavior. Taken together, these evaluations provide a comprehensive picture of the model's readiness for real-world interaction and downstream applications.

Compared to the pretraining evaluation, we employ a mix of generation-based benchmarks, which require instruction-following capabilities to format the final answer, and probabilistic evaluations. We jointly consider English and multilingual benchmarks, and emphasize the importance of analyzing them together.

Benchmarks. We consider a suite of benchmarks in seven categories that capture complementary aspects of model capabilities. Knowledge recall includes AGIeval (Zhong et al., 2024), MMLU (Hendrycks et al., 2021a), Global-MMLU (Singh et al., 2025), TruthfulQA (Lin et al., 2022), and TruthfulQA multilingual (Calvo Figueras et al., 2025). Instruction following is evaluated with IFEval (Zhou et al., 2023) and Multi-IFEval (Dussolle et al., 2025), and Commonsense reasoning is covered by HellaSwag (English; Zellers et al., 2019; multilingual; Lai et al., 2023). Coding abilities are tested with HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021), while the mathematical evaluation spans GSM8K (Cobbe et al., 2021), GSM8K-Platinum), MATH (Hendrycks et al., 2021b), and MathQA (Amini et al., 2019). To assess the reasoning capabilities of the models, we use ACPBench (Kokel et al., 2025), ARC Challenge (Clark et al., 2018), BBH (Suzgun et al., 2023), DROP (Dua et al., 2019), GPQA (Rein et al., 2024), MGSM (Shi et al., 2023), and MLogiQA (Liu et al., 2021). We further include a broad set of benchmarks evaluating cultural knowledge, including BLEnD (Myung et al., 2024), CulturalBench (Chiu et al., 2025), INCLUDE (Romanou et al., 2025), and our custom SwitzerlandQA (§F.7). We provide details on the benchmark specifications in Table 32. Benchmarks contained in Table 31 were held-out during model development and were not used for making decisions.

Baseline Models. We compare our models against a range of instruction-tuned baselines, spanning both open-weight and fully open-source models with parameter sizes from 3B to 72B. These baselines include model families such as LLaMA, Qwen, OLMo, EuroLLM, and Gemma. The complete list of models is provided in Table 26.

Evaluation Setup. Consistent with the evaluation approach used during pretraining, we employ the *lm-evaluation-harness* framework in the post-training phase, shifting to open-generation mode to better assess the model's generative capabilities. We rely on the framework's existing benchmark implementations while extending it with additional tasks not natively supported, carefully adhering to the original task definitions, prompt formats, and evaluation protocols specified in their respective publications. To ensure methodological fairness and consistency, particularly when evaluating smaller models, we adopt simplified prompting strategies and apply additional extraction filters to standardize response parsing and improve evaluation reliability. Moreover, we continue to track the model's pretraining competencies throughout post-training (see Section F.1), extending probabilistic evaluation of pretraining benchmarks to zero-shot

and zero-shot chain-of-thought (CoT) generation. This enables a more nuanced analysis of how foundational skills evolve under alignment.

Post-training Evaluation Results. Evaluation results are presented across different capability categories: Knowledge recall, Instruction following, and Commonsense reasoning in Table 27; Coding and Math in Table 28; Reasoning in Table 29; and Cultural knowledge in Table 30. Results on the held-out test suite spanning Knowledge, Reasoning, and Math are reported in Table 31.

Overall, comparisons between models on development metrics align well with results from the held-out evaluation suite (Table 31). The Apertus-Instruct models achieve solid performance across the diverse set of benchmarks considered, particularly in comparison to other fully open models of similar sizes. Notably, Apertus-8B is competitive with the strongest fully open models in knowledge recall, instruction following, and commonsense reasoning, while performing less strongly in math, coding, and reasoning. At the same time, it stands out in cultural knowledge, where it leads among fully open models and approaches the strongest models in its size class, such as Qwen3-8B. Performance in math and coding is comparatively weaker for both Apertus models, though most other models have undergone additional RL training (*e.g.*, RLVR), which is known to enhance these capabilities but has not yet been applied to Apertus. The performance gap between the 8B and 70B models is smaller than typically observed in other model families.

Long Context Evaluation. We evaluate the long context capabilities of Apertus-8B-Instruct and Apertus-70B-Instruct on the RULER (Hsieh et al., 2024) benchmark with a configured context length of 4k, 8k, 16k, 32k, and 64k. The evaluation results are shown in Table 42.

F.3 Low-resource Translation

As our model is pretrained on low-resource languages, we specifically test Apertus's translation abilities to and from Romansh, a low-resource language that is one of Switzerland's four official languages. We use the Romansh WMT24++ benchmark for machine translation (Vamvas et al., 2025), which evaluates translation quality between German and either of six written varieties of the Romansh language – Rumantsch Grischun as well as the regional varieties Sursilvan, Sutsilvan, Surmiran, Puter, and Vallader. The benchmark is an extension of WMT24++ (Deutsch et al., 2025) and follows the protocol of the WMT24 General Machine Translation Shared Task (Kocmi et al., 2024), *i.e.*, few-shot prompting with 3 example sentence pairs and greedy decoding. Table 43 reports the BLEU score (Papineni et al., 2002) of the generated translations. Across the board, Apertus-70B-Instruct demonstrates greater low-resource translation abilities compared to Llama-3.3-70B-Instruct.

F.4 Memorization Prevention

Robustness Across Decoding Strategies. Prior work established a connection between memorization and repetition-induced text degeneration (Xu et al., 2025), a phenomenon also observed for Apertus under greedy decoding (Table 44). TTR values remain low (0.22–0.31), increasing with exposure frequency but still well below the ground truth ($\sim$0.539). Qualitative inspection suggests this stems from thematic loops, particularly for rarely or unseen texts, which can produce artificially low Rouge-L scores ($\sim$0.18) reflecting poor generation quality rather than genuine mitigation. To rule this out, we also evaluate nucleus sampling (`temperature=1.0`, `top-p=0.9`). Under this setting, Apertus maintains a high TTR ($\approx$0.500) close to the ground truth, while Rouge-L and LCCS remain at baseline. These results confirm that Apertus's mitigation is robust across decoding strategies and not an artifact of greedy decoding.

Goldfish Loss Alters Memorization Dynamics. Prior work has shown the positional fragility of LLM memorization under standard cross-entropy and full attention: initial tokens in the context window trigger the strongest recall, while memorization decays as prefixes shift further away (Xu et al., 2025). Our findings suggest that the Goldfish loss breaks this dependency: selective token masking disrupts the formation of continuous long-range anchors on early tokens, which otherwise facilitate verbatim memorization. For the top 5% of most-memorized sequences (after filtering as in §F.4.1), recall does not follow the sharp offset-dependent decay predicted by positional fragility in Xu et al. (2025). Instead, it fluctuates within a narrow range (Figure 11), and the specific sequences vary with offset, likely because deterministic masking exposes different "unprotected" windows at different positions.

Potential Primacy Effect. Figure 11 also suggests a potential primacy effect: Gutenberg sequences introduced during the first 0–9T tokens of pretraining appear more strongly memorized than those introduced in 9–12T. This pattern, however, may be confounded by differences in textual complexity between the v1 and v2 Gutenberg probe sets and therefore warrants further investigation.

F.4.1 Failure Case Studies

Despite its success, Goldfish Loss has a key limitation: its deterministic hashing is fragile to near-duplicates. This property becomes critical when training data contains multiple, slightly varied versions of the same text. Our analysis shows that the most frequently memorized sequences are overwhelmingly canonical works, including Keats's poems, Shakespeare's plays, the US Constitution, and the Bible, which appear both in our Gutenberg sequences and repeatedly in the 15T pretraining corpus, accounting for all 22 sequences with a ROUGE-L score ≥ 0.7 among our 10,672 Gutenberg probes.

Goldfish Loss hashes a fixed-size preceding context ($H = 50$ tokens) to decide which tokens to mask, but even small divergences alter the hash. We identify two main sources: (i) **Formatting divergence**, since our Gutenberg sequences follow a fixed layout of ~ 21.5 tokens per line, whereas web versions often differ in line-breaking, introducing varying numbers of \n tokens; and (ii) **Tokenizer inconsistency**, where leading whitespace or subword segmentation produces different token IDs (Bostrom and Durrett, 2020; Chai et al., 2024). A single-token shift is enough for Gutenberg and web variants of the same passage to be masked inconsistently, so tokens masked in the Gutenberg version are revealed in the web version, allowing the model to memorize the entire sequence.

We also find "false positives" as shown in Figure 12: high verbatim recall of structured, low-diversity content (*e.g.*, tables, recipe lists, contents pages). Here, high ROUGE-L reflects template learning rather than true verbatim memorization, typically on true suffixes with TTR ≤ 0.4 for a 500-token suffix. Such cases carry lower copyright and privacy risks than memorization of literary passages.

F.5 Security And Safety

F.5.1 General Considerations

As a highly multilingual, fully open model, the safety and security testing of the Apertus model family presents several unique challenges.

Open-weight. As an open-weight model family, any security and safety guardrails imparted into the model during pretraining can be reverted through post-training (*e.g.*, Team 2025). Hence, we cannot assume that access to potentially dangerous information acquired by the model from the pretraining data can be mitigated through safety alignment alone. As a result, we already implemented data compliance measures (*e.g.*, author opt-outs, PII filtering, toxicity filtering), *a priori*, during pretraining data construction (§2).

Massively Multilingual. As a highly multilingual model family, Apertus's security and safety should be maintained across supported languages. This task is challenging, given that most safety and security work focuses almost exclusively on English, resulting in poor generalization to other languages (Wang et al., 2024), and in translations serving as effective jailbreaks (Deng et al., 2024; Yong et al., 2023). Consequently, we test the safety of our model on available multilingual safety benchmarks (Ning et al., 2025), but still fall short on all languages used in our pretraining and post-training datasets. An additional challenge with massively multilingual models is their novel capacity for information operations in low-resource languages (Kucharavy et al., 2023; Goldstein et al., 2023). Consequently, we conducted manual tests for several high-risk scenarios (§F.6).

Helpfulness vs. Safety. As the Apertus models are intended for wide adoption, they must be useful to broad communities of users. Given that there is a trade-off between model harmlessness and usefulness after tuning (Bai et al., 2022a,b; Röttger et al., 2024), an excessive safety and security emphasis is likely to impede the model utility. This trade-off also means that potentially harmful behaviours are impossible to suppress without making the model useless for certain applications. Consequently, we seek a balance in our development between these two properties. Notably, given the post-training guardrail removal risk mentioned above, we do not pursue jailbreak resistance, given that it must be delegated to guardrails in production (Majumdar and Vogelsang, 2024).

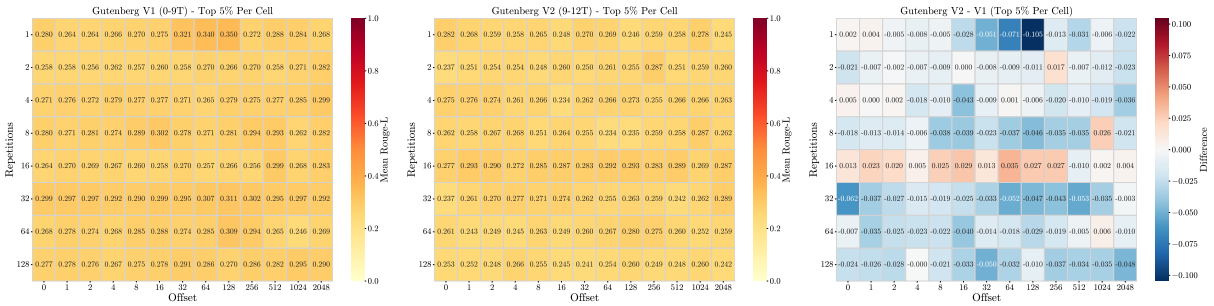

Figure 11: **Temporal and Altered Positional Memorization Dynamics.** The heatmaps compare memorization for Gutenberg-V1 sequences (injected into the first 9T tokens of pretraining) versus Gutenberg-V2 sequences (injected between the 9-12T token marks) for the top 5% most-memorized sequences, evaluated using 500-token prefixes to generate 500-token suffixes. The x-axis represents the offset—the number of tokens skipped from the start of a sequence before prefix extraction—varied from 0 to 2048. The rightmost plot (V2 - V1) is predominantly blue, indicating that sequences from the earlier training stage (V1) were more strongly memorized (a primacy effect). The difference can be substantial; for instance, a Rouge-L difference of 0.1, as seen in some cells, corresponds to 50 additional tokens being memorized in the 500-token suffix. Both the V1 and V2 plots show that for the top memorized sequences, recall fluctuates across offsets rather than exhibiting the sharp decay characteristic of positional fragility.

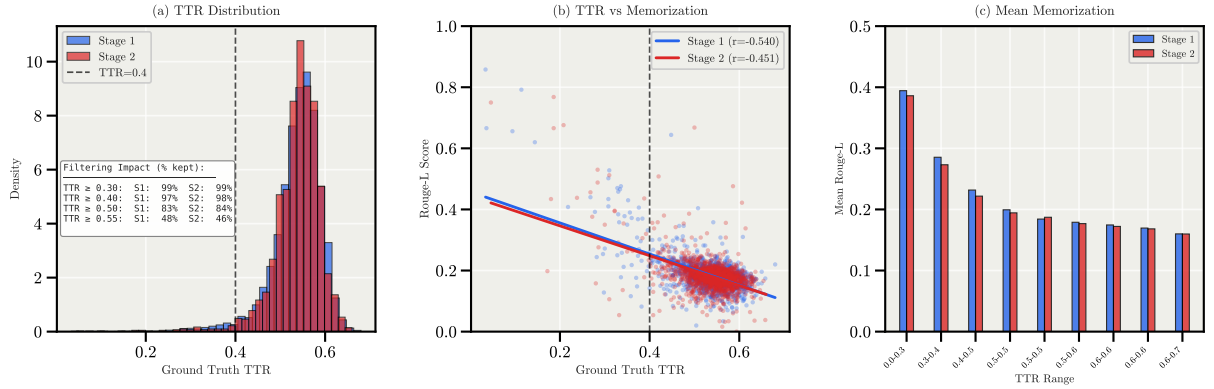

Figure 12: **Memorization patterns across TTR distributions for 500-token suffixes.** (a) Distribution of ground truth TTR values for Stage 1 (500 sequences per bucket) and Stage 2 (167 sequences per bucket). The vertical line at TTR=0.4 marks the threshold below which sequences are predominantly structured, repetitive content. (b) Negative correlation between TTR and ROUGE-L scores ($r = -0.540$ for Stage 1, $r = -0.451$ for Stage 2), demonstrating that low-diversity sequences exhibit higher verbatim recall. (c) Mean memorization levels across TTR ranges, confirming that sequences with TTR ≤ 0.4 show elevated ROUGE-L scores, often representing template learning rather than true verbatim memorization of unique content.

F.5.2 Safety Benchmark Performance

Based on the principles outlined above, we perform safety testing using the following benchmarks:

BBQ is an English-language common harmful social bias evaluation benchmark (Parrish et al., 2022). It is constructed to elicit implicit biases on common discrimination categories (*e.g.*, Age, Disability, Gender, Ethnicity, etc.), probing for bias in question-answers known to elicit harmful bias. We observe that the Apertus-Instruct family performs comparably to other fully-open models, though a bit worse than state-of-the-art open-weight models.

HarmBench is a standardized LLM harmful behaviour elicitation benchmark, covering 8 classes of harmful behaviour (Bioweapon, Harassment, General Harm, Chemweapon, Cybercrime, Misinformation, Copyright, Illegal Act; Mazeika et al., 2024). On HarmBench Direct Requests, we observe the Apertus-Instruct family performing comparably to other fully-open models and better than most open-weight models tested (with the exception of Qwen). Including human jailbreaks, the most basic approaches to LLM jailbreaking, also indicates a performance of the Apertus-Instruct family in line with most open-weight models tested (excluding Qwen).

RealToxicityPrompts is one of the most widely used benchmarks for unprompted toxicity generation in the LLMs, considered as representative of real-world usage scenarios in English (Gehman et al., 2020). To integrate it in our benchmark harness, we sub-sample it to 10% of its size and switch the toxicity classifier model to Llama-Guard-3-8B (Fedorov et al., 2024) to allow fully-contained execution. The resulting benchmark, *RealToxicityPrompts-Llama-Subsampled*, while quicker for evaluation, cannot be directly compared with the standard *RealToxicityPrompts* benchmark results. Overall, we observe that Apertus models perform well in comparison to other both fully open and open-weight models.

ToxiGen is an English benchmark for evaluating the implicit toxicity of LLM generations, as well as the ability of a model to identify that implicit toxicity (Hartvigsen et al., 2022). We use the version of ToxiGen for evaluating the ability of a model to accurately identify implicit toxicity on a balanced dataset. Overall, we observe that the family of Apertus-Instruct models is in line with the rest of the fully-open models tested, but performs worse than all open-weight models tested.

LinguaSafe is a recent multilingual LLM safety benchmark (Ning et al., 2025) across 5 classes and 12 languages: (1) *Crimes*, (2) *Explicit Content*, (3) *Fairness*, (4) *Harm*, and (5) *Privacy*. This benchmark separates detected harmful responses by harm class and language, and includes several mid- and low-resource languages. While Ning et al. (2025) do not report direct evaluation of security-weighted scores (as we do in this work), the direct and indirect mean weighted scores are in the range of 21-45% for open-weight models.

F.6 Qualitative Spot-Testing

Given the performance of the Apertus models on standard benchmarks was in-line with other open models, we also focused on spot-testing for test cases known to be difficult for LLMs. Specifically, we spot-test for inherently dangerous responses and common usage harms using relatively recently reported issues on state-of-the-art models in the wild.

We conducted manual testing on the released Apertus-8B-Instruct and Apertus-70B-Instruct models, notably focusing on CBRNE, Dual Use, Medical Disinformation, Private Person Claims and Suitability for Information Operations in Low-resource Languages. While we found potential for improvement in future model releases, we did not find any issue that would have warranted the delay of the model release.

F.7 SwitzerlandQA

To evaluate whether Apertus can serve as a knowledgeable foundation for Sovereign AI initiatives, we test the model on its understanding of Switzerland's environment by developing a novel benchmark **SwitzerlandQA** specifically tailored to Switzerland's context. The benchmark spans five domains that reflect the themes emphasized in Swiss naturalisation exams, civic education materials, and integration resources: *Geography*, *History*, *Social Life*, *Political Life*, and *Insurance*. *Geography* covers Switzerland's location, terrain, hiking regions, population, general economy, and climate. *History* addresses significant events, notable figures, and key developments across time. *Social Life* includes food, traditions, and festivals that shape everyday culture. *Political Life* focuses on Switzerland's political organization and institu-

tional structure. *Insurance*, a particularly relevant civic topic, covers national insurance rules as well as canton-specific systems for subsidies and related regulations.

The benchmark represents 26 cantons, with each canton having at least 200 questions, yielding 9,167 unique items per language across domains and levels of granularity. To support multilingual evaluation, the dataset was translated into German, French, Italian, Romansh, and English. Automatic translations were subsequently sampled and checked to ensure semantic fidelity and terminological consistency. Each item follows a four-option multiple-choice format with a single correct answer.

Questions are organized at three levels of granularity:

- **National level:** broad knowledge relevant to Switzerland as a whole (*e.g.*, "What is the highest mountain in Switzerland?").
- **Cantonal level:** knowledge specific to an individual canton (*e.g.*, "What event led to Appenzell's excommunication in the early 15th century?").
- **Commune level:** fine-grained local knowledge at the municipal level (*e.g.*, "What was the population of the commune of St. Sulpice in 2023?").

Although all three levels are represented, the cantonal level was prioritized, as it provides a balance between the generality of national questions and specificity of commune ones.

For data collection, we relied primarily on official sources. Where possible, we included questions from naturalisation exams, which represent a standardized and widely recognized measure of civic knowledge. To expand coverage, we contacted cantonal cultural departments and requested resources containing authoritative information in the target domains. Where insufficient questions were available, new items were generated in English using GPT-4o, guided by official materials to preserve factual accuracy and regional relevance. In total, we compiled information from 107 unique official data sources, ranging from canton-level handbooks and civic guides to municipal archives.

The dataset also has limitations. First, while translations into German, French, Italian, and English were generally reliable, the Romansh subset carries a higher risk of translation inaccuracies due to the limited linguistic resources. Second, coverage across cantons is uneven: some cantons provide extensive official documentation on their history, culture, and institutions, while others offer much less. As a result, the depth of representation varies across cantons, which may introduce regional evaluation imbalances.

Category	Dataset Source	# Examples	Data Ratio
Foundation	OLMo2 WildChat	298,556	9.56%
	OLMo2 Personas	29,356	
	OLMo2 SciRiff	29,809	
	OLMo2 TableGPT	24,803	
	OLMo2 CoCoNot	10,793	
	OLMo2 OASST1	7,047	
	Subtotal	*400,364*	
Math & Reasoning	Llama-Nemotron Math	200,000	11.60%
	Tulu3 Personas Math (filtered)	125,522	
	NuminaMath (tool-extracted)	63,248	
	OLMo2 OpenMath GSM8K	49,948	
	Llama-Nemotron Chat/Safety	46,808	
	Subtotal	*485,526*	
Code & Functions	Llama-Nemotron Code	200,000	9.02%
	Glaive Function Calling	112,688	
	XLam Function Calling	60,000	
	APIGen	5,000	
	Subtotal	*377,688*	
Multilingual	SmolTalk2 (8 languages)	1,273,789	34.22%
	EuroBlocks Multilingual	157,318	
	s1k_42_langs (filtered)	1,000	
	Subtotal	*1,432,107*	
Regional	WikiQA	883,513	22.54%
	Romansh	46,170	
	Swiss-German Dialects	6,179	
	African Languages	7,339	
	Swiss Charter Q&A	226	
	Subtotal	*943,427*	
Domain-Specific	The-Tome (Financial/Web)	544,975	13.02%
Total		**4'184'087**	100%

Table 20: **SFT data mixture composition by source and category**. All datasets are decontaminated against evaluation benchmarks. Numbers indicate example count after filtering.

Idiom / Split	Dictionary	Sentences	Idiom labels	Total
Rumantsch Grischun (RG)	14,264	1,038†	3,000	**18,302**
Surmiran	7,486	198	3,000	**10,684**
Sursilvan	1,352	182	3,000	**4,534**
Sutsilvan	5,854	–	1,322	**7,176**
Vallader	–	88	3,000	**3,088**
Puter	–	–	3,000	**3,000**
Human translated	–	–	–	**139**
Total				**46,923**

Table 21: Romansh SFT dataset counts (examples). Dictionary and sentence splits are bidirectional in the final SFT where both directions were generated; numbers reflect the released SFT splits. †RG sentence breakdown (released totals): de↔RG 234, en↔RG 262, fr↔RG 276, it↔RG 266.

Principle	Response Categories (%)					Agree / Agree+Disagree
	Always/ definitely not	Usually/ probably not	Neutral/ Unsure	Usually/ probably yes	Always/ definitely yes	
1. Response Quality	0.5	0.0	6.5	17.8	75.2	**99.4**
2. Knowledge and Reasoning Standards	0.0	0.5	2.7	9.7	87.1	**99.4**
3. Respectful Communication	0.5	3.2	4.9	21.1	70.3	**95.4**
4. Preventing Harm	0.0	1.1	1.1	6.5	91.3	**98.9**
5. Resolving Value Conflicts	0.0	1.6	5.9	24.9	67.6	**97.5**
6. Professional Competence Boundaries	0.5	5.4	6.0	26.3	61.8	**92.6**
7. Collective Decision-Making	0.0	4.9	7.6	26.5	61.0	**94.9**
8. Autonomy and Personal Boundaries	0.5	3.3	5.5	18.1	72.6	**96.4**
9. Long-term Orientation and Sustainability	0.5	3.8	9.7	26.5	59.5	**93.6**
10. Human Agency	0.5	2.2	6.0	21.1	70.2	**96.7**
11. AI Identity and Limits	0.0	3.3	8.2	22.4	66.1	**95.8**
Average	0.3	2.7	5.7	19.0	71.8	**97.3**

Table 22: **Survey Approval on Values Expressed in Swiss AI Charter**. Rows correspond to the 11 articles of the Swiss AI Charter The five middle columns correspond to answers to the main survey question: *"When interacting with human users, to what extent should AI chatbots follow this principle?"*. The rightmost column is the sum of the "yes" answers divided by the sum of the "yes" and "no" answers (excluding "neutral"). The bottom row is the column average. All numbers in percent.

Table 23: Accuracy by Prompt. Best metric for a model is in bold.

Model	Prompt 1	Prompt 2	Prompt 3	Prompt 4
Qwen3-32B	0.673	**0.734**	0.682	0.715
Llama-3.3-70B-Instruct	**0.720**	0.715	0.598	0.607
Qwen2.5-VL-72B-Instruct	0.710	**0.724**	0.593	0.706
DeepSeek-R1-0528	**0.714**	0.656	0.606	0.623

		General Language Understanding					
Model	**Avg**	**ARC** (↑)	**HellaSwag** (↑)	**WinoGrande** (↑)	**XNLI** (↑)	**XCOPA** (↑)	**PIQA** (↑)
Fully Open Models							
Apertus-8B	65.8	72.7	59.8	70.6	45.2	66.5	79.8
Apertus-70B	67.5	70.6	64.0	73.3	45.3	69.8	81.9
OLMo2-7B	64.0	72.9	60.4	74.5	40.4	55.2	80.9
OLMo2-32B	67.7	76.2	66.7	78.6	42.9	60.1	82.1
EuroLLM-1.7B	54.8	57.2	44.9	58.1	40.7	55.7	72.4
EuroLLM-9B	62.8	67.9	57.9	68.8	41.5	61.1	79.6
SmolLM2-1.7B	58.5	66.1	52.4	65.6	37.6	52.3	77.0
SmolLM3-3B	61.6	68.6	56.4	68.1	40.5	58.2	77.7
Poro-34B	61.7	65.7	57.9	70.6	41.6	56.0	78.5
Open-Weight Models							
Llama3.1-8B	65.4	71.6	60.0	73.4	45.3	61.8	80.1
Llama3.1-70B	67.3	74.4	56.5	79.4	44.3	66.7	82.3
Qwen2.5-7B	64.4	69.6	60.1	72.8	43.3	61.7	78.7
Qwen2.5-72B	69.8	76.2	67.5	78.0	46.9	68.2	82.0
Qwen3-32B	67.8	75.6	64.0	73.8	44.4	67.9	80.9
Llama4-Scout-16x17B	67.9	74.7	66.8	73.2	43.5	67.7	81.2
GPT-OSS-20B	58.1	67.0	41.5	66.5	37.4	60.4	75.6

Table 24: **Pretraining Evaluation:** Performance (%) of Apertus models on *general language understanding* tasks compared to other pretrained models. The arrows (↑,↓) show the desired direction for each benchmark.

		Factual Agnostic		Factual Regional				
Model	**Avg**	**MMLU** (↑)	**Global-MMLU** (↑)	**INCLUDE V1** (↑)	**INCLUDE V2** (↑)	**Cultural-Bench** (↑)	**BLEND** (↑)	**SwitzerlandQA** (↑)
Fully Open Models								
Apertus-8B	56.9	61.6	55.3	54.8	37.3	55.2	72.2	62.1
Apertus-70B	58.9	65.2	58.2	57.0	38.5	58.1	75.0	60.2
OLMo2-7B	51.6	60.5	41.1	33.8	30.6	69.5	73.2	52.5
OLMo2-32B	62.0	71.9	57.4	50.6	37.5	74.8	79.4	62.4
EuroLLM-1.7B	26.3	25.4	26.2	24.5	26.2	31.5	24.4	25.9
EuroLLM-9B	47.7	55.0	46.6	43.0	32.7	47.0	51.7	58.1
SmolLM2-1.7B	35.3	47.2	31.6	27.6	28.4	65.7	24.4	22.4
SmolLM3-3B	49.7	59.7	48.5	39.0	31.5	56.5	57.5	55.2
Poro-34B	37.5	44.3	34.8	31.0	26.8	40.2	43.4	42.1
Open-Weight Models								
Llama3.1-8B	53.2	63.4	52.1	48.8	37.4	43.1	68.9	58.5
Llama3.1-70B	66.7	75.9	69.8	64.1	43.7	62.3	82.4	68.6
Llama4-Scout-16x17B	67.0	75.4	70.2	67.3	46.3	56.4	81.1	72.0
Qwen2.5-7B	58.6	71.9	60.3	53.9	37.8	47.3	75.2	63.9
Qwen2.5-72B	72.5	83.3	77.0	69.7	44.5	76.8	83.4	72.7
Qwen3-32B	69.1	80.7	71.1	67.7	41.8	74.9	81.0	66.5
GPT-OSS-20B	58.1	56.6	57.7	43.5	40.2	66.2	77.0	65.3

Table 25: **Pretraining Evaluation:** Performance (%) of Apertus models on *factual knowledge acquisition* tasks compared to other pretrained models. The arrows (↑,↓) show the desired direction for each benchmark.

Model	Open-weight	Fully-open	Multilingual Focus
Pretrained Baselines			
OLMo2-7B (Walsh et al., 2025)	✓	✓	✗
OLMo2-32B (Walsh et al., 2025)	✓	✓	✗
EuroLLM-1.7B (Martins et al., 2024)	✓	✓	✓
EuroLLM-9B (Martins et al., 2024)	✓	✓	✓
SmolLM2-1.7B (HuggingFaceTB, 2025)	✓	✓	✓
SmolLM3-3B (HuggingFaceTB, 2025)	✓	✓	✓
Llama3.1-8B (Grattafiori et al., 2024)	✓	✗	✓
Llama-3.3-70B (Grattafiori et al., 2024)	✓	✗	✓
Llama4-Scout-16x17B (Meta AI, 2025)	✓	✗	✓
Qwen2.5-7B (Yang et al., 2025)	✓	✗	✓
Qwen2.5-72B (Qwen et al., 2024)	✓	✗	✓
Qwen3-32B (Yang et al., 2025)	✓	✗	✓
GPT-OSS-20B (OpenAI et al., 2025)	✓	✗	✓
Post-trained Baselines			
ALLaM-7B-Instruct-preview (Bari et al., 2025)	✓	✓	✓
EuroLLM-22B-Instruct-Preview (Martins et al., 2024)	✓	✓	✓
EuroLLM-9B-Instruct (Martins et al., 2024)	✓	✓	✓
K2-Chat (Liu et al., 2025b)	✓	✓	✓
Llama-3.1-8B-Instruct (Grattafiori et al., 2024)	✓	✗	✓
Llama-3.3-70B-Instruct (Grattafiori et al., 2024)	✓	✗	✓
gemma-3-12b-it (Kamath et al., 2025)	✓	✗	✓
gemma-3-27b-it (Kamath et al., 2025)	✓	✗	✓
marin-8b-instruct (Community, 2025)	✓	✓	✓
Minerva-7B-instruct-v1.0 (NLP, 2024)	✓	✓	✓
OLMo-2-0325-32B-Instruct (Walsh et al., 2025)	✓	✓	✗
OLMo-2-0325-32B-SFT (Walsh et al., 2025)	✓	✓	✗
OLMo-2-1124-7B-Instruct (Walsh et al., 2025)	✓	✓	✗
OLMo-2-1124-7B-SFT (Walsh et al., 2025)	✓	✓	✗
Qwen2.5-72B-Instruct (Qwen et al., 2024)	✓	✗	✓
Qwen3-32B (Yang et al., 2025)	✓	✗	✓
Qwen3-8B (Yang et al., 2025)	✓	✗	✓
salamandra-7b-instruct (Gonzalez-Agirre et al., 2025)	✓	✓	✓
SmolLM3-3B (HuggingFaceTB, 2025)	✓	✓	✓
Teuken-7B-instruct-v0.6 (Ali et al., 2024)	✓	✓	✓

Table 26: **Pretrained and Post-trained Baseline LLMs**, compared with Apertus and Apertus-Instruct. **Fully-open** indicates whether the models provide open data, open weights, and open implementations.

Model	Avg (↑)	Knowledge				Commonsense Reasoning	
		MMLU (↑)	Global-MMLU (↑)	TruthQA (↑)	TruthQA Multilingual (↑)	HellaSwag (↑)	HellaSwag Multilingual (↑)
Fully Open Models							
Apertus-70B-Instruct	63.4	69.6	62.7	61.2	53.7	78.1	55.3
Apertus-8B-Instruct	58.8	60.9	55.7	56.7	52.4	74.6	52.7
ALLaM-7B-Instruct-preview	53.7	62.9	50.6	47.5	43.7	75.3	42.0
EuroLLM-22B-Instruct-Preview	58.3	65.3	56.9	56.6	49.8	73.0	48.1
EuroLLM-9B-Instruct	53.8	58.4	52.0	49.7	46.5	69.8	46.3
K2-Chat	56.8	65.7	49.8	56.5	49.2	74.9	44.7
marin-8b-instruct	54.5	65.5	48.4	55.2	47.6	72.0	38.1
Minerva-7B-instruct-v1.0	40.8	30.7	28.5	44.0	47.2	63.3	31.2
OLMo-2-0325-32B-Instruct	68.0	77.9	61.3	73.2	56.4	86.0	53.0
OLMo-2-1124-7B-Instruct	53.7	60.0	42.8	56.5	46.5	77.5	38.7
salamandra-7b-instruct	52.0	52.4	43.1	51.0	48.4	71.4	45.9
SmolLM3-3B	54.4	61.7	51.2	54.3	50.0	69.0	40.4
Teuken-7B-instruct-v0.6	48.9	49.0	39.9	46.4	48.1	67.8	42.2
Open-Weight Models							
gemma-3-12b-it	60.8	78.8	69.6	60.8	56.1	53.7	45.6
gemma-3-27b-it	63.8	83.6	75.3	64.4	54.8	54.9	49.8
Llama-3.1-8B-Instruct	59.2	72.4	57.1	55.1	50.8	72.5	47.0
Llama-3.3-70B-Instruct	68.4	87.5	77.8	66.1	55.2	70.1	53.8
Qwen2.5-72B-Instruct	68.8	86.6	77.7	69.7	58.6	68.8	51.5
Qwen3-32B	64.1	83.7	74.8	58.6	50.7	68.8	48.0
Qwen3-8B	57.8	79.1	64.0	53.4	51.4	58.6	40.4

Table 27: **Post-training Evaluation:** Performance (%) of Apertus models across **knowledge recall**, and **commonsense reasoning**. Performance is reported on benchmarks for both English and multilingual settings. The arrows (↑,↓) show the desired direction for each benchmark.

Model	Avg (↑)	Coding		Math			
		HumanEval (Pass@10) (↑)	MBPP (Pass@1) (↑)	GSM8K (↑)	MGSM (↑)	Hendrycks Math (↑)	MathQA (↑)
Fully Open Models							
Apertus-70B-Instruct	54.4	73.0	47.0	77.6	64.3	30.8	33.9
Apertus-8B-Instruct	44.2	67.0	36.2	62.9	48.5	18.2	32.1
ALLaM-7B-Instruct-preview	38.5	56.7	39.0	58.2	29.1	15.6	32.3
EuroLLM-22B-Instruct-Preview	53.0	75.2	43.0	75.5	50.7	38.0	35.4
EuroLLM-9B-Instruct	42.9	65.3	41.0	62.9	36.1	19.2	32.7
K2-Chat	59.5	87.7	56.2	84.8	49.1	40.7	38.7
marin-8b-instruct	51.7	85.8	41.2	80.6	42.8	31.3	28.6
Minerva-7B-instruct-v1.0	14.5	25.0	17.2	13.6	2.8	3.5	24.7
OLMo-2-0325-32B-Instruct	56.7	69.0	41.8	88.2	67.3	44.3	29.6
OLMo-2-1124-7B-Instruct	45.8	65.2	32.0	83.5	36.9	31.1	26.0
salamandra-7b-instruct	19.4	28.4	22.2	22.7	9.6	5.2	28.6
SmolLM3-3B	58.5	89.7	52.8	83.6	45.2	51.8	27.7
Teuken-7B-instruct-v0.6	27.7	44.6	25.6	38.1	19.2	11.4	27.1
Open-Weight Models							
gemma-3-12b-it	71.1	88.0	72.0	89.9	68.9	68.4	39.3
gemma-3-27b-it	73.1	89.3	72.8	90.4	71.7	71.1	43.1
Llama-3.1-8B-Instruct	60.0	86.7	60.6	84.5	67.7	36.3	24.4
Llama-3.3-70B-Instruct	74.3	95.8	75.6	94.8	86.0	60.3	33.5
Qwen2.5-72B-Instruct	74.6	95.4	74.6	88.6	76.2	67.8	44.8
Qwen3-32B	76.3	97.0	73.6	93.6	74.0	69.2	50.5
Qwen3-8B	68.8	95.6	66.8	89.5	52.0	66.8	41.8

Table 28: **Post-training Evaluation:** Performance of Apertus models on **coding and mathematical reasoning** tasks. The arrows (↑,↓) show the desired direction for each benchmark.

Model	Avg (↑)	Reasoning				Instruction Following	
		BBH (↑)	DROP (↑)	ACP-Bench Bool (↑)	ACP-Bench MCQ (↑)	IFEval (↑)	Multi-IFEval (↑)
Fully Open Models							
Apertus-70B-Instruct	61.8	64.2	50.8	62.9	43.0	75.2	74.7
Apertus-8B-Instruct	56.0	55.9	49.7	58.4	31.2	71.7	68.9
ALLaM-7B-Instruct-preview	53.6	46.3	55.4	58.9	41.7	65.4	54.0
EuroLLM-22B-Instruct-Preview	58.8	56.3	47.5	60.9	43.3	72.8	72.0
EuroLLM-9B-Instruct	51.3	53.1	45.0	51.6	34.0	62.8	61.3
K2-Chat	53.9	70.7	57.3	58.6	41.7	48.4	47.0
marin-8b-instruct	55.9	61.5	60.3	49.9	33.0	68.8	62.1
Minerva-7B-instruct-v1.0	27.5	28.2	29.5	44.7	23.3	19.4	19.8
OLMo-2-0325-32B-Instruct	75.1	64.1	77.9	79.0	63.1	86.0	80.6
OLMo-2-1124-7B-Instruct	55.9	50.1	60.3	57.1	36.3	71.0	60.6
salamandra-7b-instruct	37.7	43.6	37.5	49.7	28.2	33.6	33.7
SmolLM3-3B	59.9	68.4	47.3	63.2	38.1	72.3	70.1
Teuken-7B-instruct-v0.6	35.7	42.4	35.9	46.2	28.0	31.6	29.9
Open-Weight Models							
gemma-3-12b-it	75.2	70.8	70.3	77.1	73.0	80.0	80.2
gemma-3-27b-it	76.9	70.8	71.1	82.9	75.4	81.3	80.0
Llama-3.1-8B-Instruct	63.9	72.0	62.4	56.2	42.8	78.6	71.3
Llama-3.3-70B-Instruct	83.8	86.6	72.0	82.6	82.1	90.8	88.7
Qwen2.5-72B-Instruct	79.4	82.7	64.4	81.6	77.6	86.3	83.8
Qwen3-32B	80.8	86.1	65.2	85.1	77.1	87.2	84.4
Qwen3-8B	73.3	53.6	60.6	82.1	74.2	86.5	82.8

Table 29: **Post-training Evaluation:** Performance (%) of Apertus models on general and logical **reasoning** tasks and **instruction following**. The arrows (↑,↓) show the desired direction for each benchmark.

Model	Avg (↑)	Cultural Knowledge				
		INCLUDE (↑)	INCLUDE V2 (↑)	BLEnD (↑)	Cultural Bench (↑)	Switzerland QA (↑)
Fully Open Models						
Apertus-70B-Instruct	61.5	58.2	41.6	66.3	74.2	67.2
Apertus-8B-Instruct	58.6	54.3	39.2	63.6	72.8	63.1
ALLaM-7B-Instruct-preview	55.2	44.4	34.6	66.4	74.4	56.0
EuroLLM-22B-Instruct-Preview	57.0	53.7	36.0	63.6	70.2	61.6
EuroLLM-9B-Instruct	54.3	49.3	36.8	62.7	61.4	61.2
K2-Chat	56.3	44.3	33.8	68.2	73.3	62.0
marin-8b-instruct	52.5	38.9	34.4	61.9	73.4	53.7
Minerva-7B-instruct-v1.0	39.1	25.6	28.0	40.4	64.0	37.4
OLMo-2-0325-32B-Instruct	58.1	52.9	39.5	61.2	74.5	62.2
OLMo-2-1124-7B-Instruct	49.7	36.3	31.3	60.8	72.8	47.2
salamandra-7b-instruct	52.8	42.1	33.0	58.6	70.5	59.6
SmolLM3-3B	52.7	41.4	31.1	61.6	72.6	56.6
Teuken-7B-instruct-v0.6	49.7	39.7	31.3	53.8	70.7	53.0
Open-Weight Models						
gemma-3-12b-it	63.4	62.7	42.8	69.5	76.8	65.1
gemma-3-27b-it	67.7	67.9	46.9	74.2	78.4	71.0
Llama-3.1-8B-Instruct	58.2	53.4	34.0	67.3	76.2	60.0
Llama-3.3-70B-Instruct	69.6	71.9	45.8	75.1	81.0	74.3
Qwen2.5-72B-Instruct	66.8	70.0	42.2	75.4	76.3	70.0
Qwen3-32B	65.9	70.6	45.8	72.0	75.5	65.6
Qwen3-8B	60.4	60.7	38.7	65.9	75.8	60.7

Table 30: **Post-training Evaluation:** Performance (%) of Apertus models on **cultural knowledge**, measuring cultural and factual knowledge across multiple languages. The arrows (↑,↓) show the desired direction for each benchmark.

Model	Avg (↑)	Test Evaluations					
		AGIeval (↑)	ARC Challenge Chat (↑)	ARC Challenge Multilingual (↑)	GPQA Main (↑)	GSM8K Platinum (↑)	MLogiQA (↑)
Fully Open Models							
Apertus-70B-Instruct	51.4	40.5	85.0	37.3	30.6	74.6	40.5
Apertus-8B-Instruct	45.1	38.7	77.6	36.8	27.0	61.6	29.0
ALLaM-7B-Instruct-preview	46.2	42.7	83.2	29.4	25.7	61.7	34.5
EuroLLM-22B-Instruct-Preview	50.2	39.9	86.4	33.3	29.0	77.3	35.4
EuroLLM-9B-Instruct	44.6	36.2	73.0	32.2	25.4	66.3	34.5
K2-Chat	49.7	43.5	79.1	32.6	29.9	77.8	35.5
marin-8b-instruct	47.7	36.5	82.6	25.5	29.9	79.1	32.8
Minerva-7B-instruct-v1.0	23.8	28.2	27.7	21.6	27.0	12.1	26.2
OLMo-2-0325-32B-Instruct	58.3	51.2	91.5	38.6	35.0	89.5	43.9
OLMo-2-1124-7B-Instruct	47.1	36.0	79.0	26.0	29.5	81.1	31.2
salamandra-7b-instruct	34.7	32.6	64.9	31.3	27.2	24.2	28.0
SmolLM3-3B	49.2	38.5	83.5	27.1	34.2	75.2	37.0
Teuken-7B-instruct-v0.6	36.4	33.0	63.4	26.7	25.0	39.5	31.1
Open-Weight Models							
gemma-3-12b-it	60.8	55.4	93.3	37.2	39.1	85.5	54.4
gemma-3-27b-it	63.5	61.3	93.8	39.8	45.1	86.7	54.5
Llama-3.1-8B-Instruct	50.3	38.1	83.7	32.0	28.3	78.8	40.9
Llama-3.3-70B-Instruct	65.8	54.2	95.7	42.9	59.6	84.0	58.1
Qwen2.5-72B-Instruct	64.9	64.1	96.2	39.2	46.9	87.3	55.9
Qwen3-32B	61.4	30.1	95.6	34.9	56.5	88.5	62.8
Qwen3-8B	56.0	29.9	93.3	30.2	42.6	89.4	50.4

Table 31: **Post-training Evaluation:** Performance (%) of Apertus models on **test benchmarks**. Results are reported on held-out benchmarks, with no feedback used during training or hyperparameter tuning. The arrows (↑,↓) show the desired direction for each benchmark.

Benchmark (identifier)	**Metric**	**Type**	**CoT**	**#Shots**	**Chat**	**Turns**	**#Langs**
ACP-Bench Bool (acp_bench_bool)	Exact Match	MCQ (Gen)	✓	2	✓	✓	1
ACP-Bench MCQ (acp_bench_mcq)	Exact Match	MCQ (Gen)	✓	2	✓	✓	1
AGIeval (agieval)	Acc.	MCQ (LH)	✗	0	✓	✗	2
ARC Challenge Chat (arc_challenge_chat_cot)	Exact Match	MCQ (Gen)	✓	0	✓	✗	1
ARC Challenge Multilingual (arc_multilingual)	Acc.	MCQ (LH)	✗	0	✓	✗	31
BBH (bbh)	Exact Match	Gen	✓	3	✓	✓	1
BBQ (bbq)	Acc.	MCQ (LH)	✗	0	✓	✗	1
BLEND (blend_sample)	Acc. (norm)	MCQ (LH)	✗	0	✓	✗	1
Cultural Bench (cultural_bench)	Acc. (norm)	MCQ (LH)	✗	0	✓	✗	1
DROP (drop)	F1	Gen	✗	3	✓	✓	1
Global MMLU (global_mmlu_gen_0shot)	Exact Match	MCQ (Gen)	✗	0	✓	✗	15
GPQA Main (gpqa_main_cot_zeroshot)	Exact Match	MCQ (Gen)	✓	0	✓	✗	1
GSM8K (gsm8k_cot)	Exact Match	Gen	✓	8	✓	✓	1
GSM8K Platinum (gsm8k_platinum_cot_zeroshot)	Exact Match	Gen	✓	0	✓	✗	1
HarmBench (harmbench)	Score	Gen	✗	0	✓	✗	1
HellaSwag (hellaswag)	Acc. (norm)	MCQ (LH)	✗	0	✓	✗	1
HellaSwag Multilingual (hellaswag_multilingual)	Acc. (norm)	MCQ (LH)	✗	0	✓	✗	31
Hendrycks Math (hendrycks_math)	Math Verify	Gen	✓	6	✓	✓	1
HumanEval (humaneval_instruct)	Pass@10	Gen	✗	0	✓	✗	1
IFEval (ifeval)	Acc. (prompt-level; loose)	Gen	✗	0	✓	✗	1
INCLUDE (include_base_44_gen_0shot)	Exact Match	MCQ (Gen)	✗	0	✓	✗	44
INCLUDE V2 (include_base_new_45_gen_0shot)	Exact Match	MCQ (Gen)	✗	0	✓	✗	45
MathQA (mathqa)	Acc.	MCQ (LH)	✗	0	✓	✗	1
MBPP (mbpp_instruct)	Pass@1	Gen	✗	3	✓	✓	1
MGSM (mgsm_en_cot)	Exact Match	Gen	✓	0	✓	✗	11
MlogiQA (mlogiqa_gen)	Exact Match	MCQ (Gen)	✗	0	✓	✗	10
MMLU (mmlu_flan_cot_zeroshot)	Exact Match	MCQ (Gen)	✓	0	✓	✗	1
Multi-IFEval (multi-if)	Acc. (prompt-level; loose)	Gen	✗	0	✓	✗	8
RealToxicityPrompts LlaMA-Guard3 Subsampled (realtoxicitypromptsllama)	Score	Gen	✗	0	✓	✗	1
Switzerland QA (switzerland_qa_0shot)	Exact Match	MCQ (Gen)	✗	0	✓	✗	5
ToxiGen (toxigen)	Acc.	MCQ (LH)	✗	0	✓	✗	1
TruthfulQA (truthfulqa_mc2)	Acc.	MCQ (LH)	✗	6	✓	✗	1
TruthfulQA Multilingual (truthfulqa_multilingual_mc2)	Acc.	MCQ (LH)	✗	5	✓	✗	31

Table 32: **Benchmark Specifications for Post-training valuations.** Benchmark name (with internal identifier in `lm-evaluation-harness`), evaluation metric, task type, use of chain-of-thought (CoT), number of few-shot demonstrations (#Shots), use of chat template (Chat), whether demonstrations are formatted as a multi-turn conversation (Turns), and the number of languages (#Langs). INCLUDE V2 is a beta extension of the INCLUDE benchmark covering 45 more languages. In total, our evaluation covers 94 different languages.

Model	Languages														
	ar	bn	de	en	es	fr	hi	id	it	ja	ko	pt	sw	yo	zh
Fully Open Models															
Apertus-70B-Instruct	62.4	53.8	67.9	74.4	69.2	68.5	56.0	66.2	68.0	65.8	62.7	68.6	50.6	38.5	68.5
Apertus-8B-Instruct	57.6	48.3	62.0	64.6	59.9	60.7	53.8	58.7	61.4	57.4	53.7	63.2	42.5	31.9	60.0
ALLaM-7B-Instruct-preview	62.5	33.9	55.7	67.0	61.7	60.3	35.3	57.7	57.2	46.1	42.7	58.5	35.2	31.8	53.4
EuroLLM-22B-Instruct-Preview	59.3	34.3	67.6	72.5	64.4	65.7	59.8	51.3	66.8	60.1	60.8	65.3	34.2	29.0	62.6
EuroLLM-9B-Instruct	53.4	28.3	59.5	64.1	61.0	58.6	50.7	49.9	59.8	59.3	57.3	59.5	30.8	28.5	59.2
K2-Chat	37.2	30.0	61.6	71.5	61.9	62.9	35.0	55.2	63.1	47.7	44.3	64.7	31.2	26.2	54.2
marin-8b-instruct	39.4	34.7	57.4	69.6	58.1	56.1	39.5	50.9	55.7	49.3	46.1	54.4	31.7	31.0	52.5
Minerva-7B-instruct-v1.0	28.2	25.9	27.2	32.5	30.8	29.3	26.1	26.4	31.2	29.3	29.8	28.5	27.3	26.0	28.7
OLMo-2-0325-32B-Instruct	57.8	47.7	71.1	79.8	71.5	71.3	54.7	67.0	70.5	60.0	52.2	67.7	50.7	34.6	62.9
OLMo-2-1124-7B-Instruct	32.8	35.3	53.5	61.8	52.0	51.1	33.1	43.5	46.4	39.1	39.6	48.8	31.6	30.9	43.4
salamandra-7b-instruct	33.2	24.1	55.1	59.3	56.4	52.3	31.3	45.1	57.2	38.4	38.4	53.4	29.5	30.1	43.2
SmolLM3-3B	53.6	27.9	62.7	69.0	63.0	61.7	45.3	49.9	60.5	50.5	50.6	60.3	28.3	28.8	55.3
Teuken-7B-instruct-v0.6	28.6	20.4	54.2	57.2	50.2	52.1	24.3	43.2	50.8	38.0	33.9	50.1	30.5	26.2	38.6
Open-Weight Models															
gemma-3-12b-it	67.4	62.2	72.0	78.5	76.1	75.7	67.6	71.9	76.9	72.1	67.7	75.5	61.1	47.5	72.1
gemma-3-27b-it	75.9	71.7	80.5	83.1	81.6	76.4	73.3	76.2	79.4	78.8	75.2	82.2	67.9	50.3	77.1
Llama-3.1-8B-Instruct	54.1	47.5	66.8	73.4	63.6	65.9	48.2	63.1	62.9	57.5	52.1	64.9	42.5	30.3	63.0
Llama-3.3-70B-Instruct	76.0	74.3	83.3	87.7	83.7	83.4	78.2	79.6	85.6	80.5	77.5	84.5	70.2	43.0	79.0
Qwen2.5-72B-Instruct	78.2	72.5	85.1	88.4	85.4	85.6	75.6	84.5	85.7	82.0	80.3	86.5	51.2	42.2	82.9
Qwen3-32B	76.8	70.2	81.3	85.8	80.6	82.0	73.2	77.9	80.8	82.4	79.4	80.8	53.5	37.7	80.1
Qwen3-8B	67.2	53.3	70.0	78.9	72.6	72.5	60.4	68.4	66.6	67.0	64.1	73.5	37.5	34.3	73.3

Table 33: **Global MMLU by language**

Model	Languages																												
	ar	bn	ca	da	de	es	eu	fr	gu	hi	hr	hu	hy	id	it	kn	ml	mr	ne	nl	pt	ro	ru	sk	sr	sv	ta	te	uk	vi
Fully Open Models																														
Apertus-70B-Instruct	58.6	41.0	64.9	68.9	67.2	71.5	36.7	70.1	37.3	49.3	64.6	59.5	30.3	65.1	69.4	35.6	32.0	37.3	38.6	69.0	70.5	65.9	65.9	61.9	64.7	68.8	31.9	33.2	61.9	60.6
Apertus-8B-Instruct	54.6	40.1	60.9	64.4	62.4	67.6	35.1	65.9	36.9	47.0	61.0	56.1	30.3	62.1	64.0	36.0	32.0	36.2	38.0	64.9	66.1	63.1	62.3	58.1	60.6	64.3	32.2	33.9	59.3	57.9
ALLaM-7B-Instruct-preview	58.5	30.4	46.0	48.8	52.1	59.9	28.9	59.3	31.2	33.7	39.9	35.1	28.0	50.5	54.9	29.8	29.4	30.0	30.7	51.1	57.4	45.8	48.6	39.9	40.7	50.7	29.1	29.9	41.2	43.9
EuroLLM-22B-Instruct-Preview	52.9	29.6	58.3	61.3	61.9	66.2	28.6	65.0	29.0	45.4	55.8	52.4	28.2	45.5	64.1	29.0	29.0	32.1	32.4	63.5	65.8	60.4	58.7	54.6	53.2	63.2	28.7	28.9	55.8	37.6
EuroLLM-9B-Instruct	51.9	30.1	56.0	59.0	58.9	62.5	29.3	62.2	30.3	45.2	52.9	49.9	27.6	40.0	59.9	29.1	28.7	31.7	32.1	60.6	62.7	58.0	54.6	52.7	50.0	59.7	28.2	29.2	53.6	34.7
K2-Chat	37.0	29.6	56.1	56.5	57.0	63.1	29.9	62.6	29.8	32.7	50.8	46.8	29.2	46.5	59.6	29.0	29.3	30.0	29.7	57.1	61.1	54.1	53.8	43.6	50.1	57.9	29.5	28.8	53.0	40.3
marin-8b-instruct	36.8	30.2	40.7	40.6	48.5	54.2	29.1	54.2	31.1	34.0	33.6	32.5	27.5	44.0	48.1	30.1	28.8	29.9	30.8	44.1	50.5	40.4	48.8	33.9	33.7	41.3	29.1	30.2	39.1	43.9
Minerva-7B-instruct-v1.0	29.0	28.1	33.5	30.2	32.5	41.1	28.3	39.3	28.6	27.3	28.4	28.9	28.1	31.3	53.6	28.5	28.8	27.4	28.2	30.7	37.6	30.3	31.3	29.4	29.0	31.2	28.0	28.5	29.1	30.1
OLMo-2-0325-32B-Instruct	55.7	40.2	62.6	64.2	69.2	74.8	30.3	73.0	37.5	49.1	55.3	44.4	28.1	66.5	70.4	35.4	32.4	37.4	38.7	66.2	71.4	64.4	66.0	51.0	55.9	66.4	30.3	33.9	54.9	57.6
OLMo-2-1124-7B-Instruct	38.6	29.9	42.9	41.9	50.1	56.6	28.8	57.9	30.6	32.4	34.8	30.9	27.3	47.6	50.5	29.4	29.4	29.6	30.1	44.8	53.4	43.7	48.1	34.0	35.2	43.2	28.5	29.9	37.3	41.0
salamandra-7b-instruct	34.4	29.5	58.9	58.4	58.5	65.7	33.8	63.8	28.9	30.1	55.1	50.1	28.0	45.3	61.2	30.0	29.2	29.5	28.5	59.4	63.1	57.1	56.2	52.5	55.6	58.9	28.8	29.1	53.4	38.6
SmolLM3-3B	49.2	29.4	42.5	40.4	56.2	61.0	29.2	59.4	29.1	40.7	35.3	31.9	27.9	43.3	57.6	29.7	29.0	31.1	31.6	42.7	58.8	39.2	54.0	35.0	36.5	40.8	28.5	29.3	41.1	46.2
Teuken-7B-instruct-v0.6	35.2	28.5	46.3	54.5	55.3	59.5	27.6	59.0	30.0	28.5	50.5	47.8	27.0	42.1	56.9	28.7	28.3	28.8	28.1	56.3	58.9	52.6	44.6	49.2	48.5	54.6	28.1	28.9	38.3	37.9
Open-Weight Models																														
gemma-3-12b-it	50.4	30.7	53.3	55.6	56.3	58.5	32.4	55.7	33.2	43.8	50.4	46.4	29.1	54.2	56.6	32.5	28.9	29.3	34.0	52.8	56.2	53.3	54.7	49.0	49.9	54.8	29.8	30.6	51.6	48.8
gemma-3-27b-it	54.3	33.6	58.1	61.8	60.9	63.5	34.2	62.6	35.0	46.9	57.4	51.4	29.8	58.9	61.6	34.7	29.3	32.4	35.6	59.7	60.8	59.1	60.6	54.7	56.6	60.4	30.2	32.3	55.9	54.6
Llama-3.1-8B-Instruct	47.4	36.4	54.5	55.0	57.8	61.2	33.2	60.2	34.6	43.4	48.6	48.8	29.8	54.1	58.6	33.7	31.3	35.0	34.5	58.4	59.5	54.1	55.5	47.1	48.8	56.8	31.0	32.0	51.7	52.0
Llama-3.3-70B-Instruct	58.6	40.0	62.9	63.3	64.6	69.5	34.8	68.6	37.1	50.6	59.8	57.4	30.2	63.5	67.1	35.9	32.6	37.2	38.7	68.0	70.1	63.3	64.4	56.6	59.5	66.3	31.3	33.2	61.0	59.6
Qwen2.5-72B-Instruct	56.3	41.6	59.6	60.4	64.0	66.8	30.5	66.8	37.5	48.2	55.8	49.2	29.7	64.0	65.3	34.0	31.9	36.3	37.4	63.2	66.8	58.8	61.2	55.0	55.7	61.7	30.9	32.6	57.0	58.5
Qwen3-32B	50.6	38.4	55.1	55.3	57.2	61.2	32.6	60.6	36.8	45.2	52.1	49.8	29.8	55.7	59.3	34.9	31.9	35.2	36.6	57.3	60.2	54.7	55.9	50.7	52.6	55.5	31.2	33.0	52.9	52.5
Qwen3-8B	42.7	33.0	44.9	44.4	47.7	51.4	29.3	50.9	32.8	37.0	40.9	39.6	28.7	47.3	49.1	31.9	30.3	31.4	32.5	46.5	50.9	44.7	46.9	40.8	41.4	45.0	29.7	30.2	42.8	44.8

Table 34: **HellaSwag Multilingual by language**

	Languages																				
Model	**albanian**	**arabic**	**armenian**	**azerbaijani**	**basque**	**belarusian**	**bengali**	**bulgarian**	**chinese**	**croatian**	**dutch**	**estonian**	**finnish**	**french**	**georgian**	**german**	**greek**	**hebrew**	**hindi**	**hungarian**	**indonesian**
Fully Open Models																					
Apertus-70B-Instruct	72.0	59.0	43.8	54.5	43.4	53.8	49.3	68.3	60.6	70.0	72.2	65.0	51.9	59.4	66.2	52.5	58.3	56.6	52.2	51.7	64.6
Apertus-8B-Instruct	68.9	54.6	44.5	47.0	38.8	50.7	50.0	64.9	55.4	69.5	59.2	63.2	50.3	48.9	62.2	53.9	54.6	43.6	53.0	46.3	59.0
ALLaM-7B-Instruct-preview	45.8	66.7	30.3	38.9	29.0	20.4	33.7	54.5	48.2	54.1	56.2	52.7	40.4	51.9	32.0	38.7	38.2	39.4	34.5	35.4	55.8
EuroLLM-22B-Instruct-Preview	48.7	58.2	28.5	43.1	34.6	43.0	29.1	70.9	55.3	73.2	65.5	69.6	59.5	65.7	39.2	54.1	58.3	40.1	56.5	49.7	55.5
EuroLLM-9B-Instruct	44.2	58.1	28.4	45.3	31.2	39.5	31.4	65.3	52.6	69.9	61.1	61.8	46.9	50.7	38.2	45.6	49.9	35.9	51.9	46.8	46.3
K2-Chat	46.9	39.3	21.6	36.7	30.8	42.2	30.7	60.8	49.3	66.5	59.1	39.5	34.7	52.1	30.6	54.0	34.1	38.0	34.6	44.0	51.0
marin-8b-instruct	33.8	38.2	29.9	31.1	30.4	25.2	32.7	37.5	45.1	37.5	43.2	24.4	38.4	51.0	31.0	48.5	34.9	37.6	38.8	32.6	49.3
Minerva-7B-instruct-v1.0	22.7	19.5	20.2	28.0	26.2	31.8	28.7	27.9	21.8	22.3	24.3	23.3	30.7	27.2	28.8	27.6	20.4	26.1	24.3	26.5	27.4
OLMo-2-0325-32B-Instruct	62.5	59.3	28.2	42.6	31.0	42.8	47.6	61.3	53.2	65.1	71.4	51.2	53.3	53.8	32.4	48.2	48.1	51.2	55.4	44.5	62.5
OLMo-2-1124-7B-Instruct	31.8	37.8	29.3	34.7	29.4	25.2	34.8	33.7	39.0	38.5	41.9	38.6	39.0	44.0	30.6	52.1	29.0	35.1	34.9	28.7	43.4
salamandra-7b-instruct	40.5	32.4	29.0	31.7	38.2	31.0	30.0	56.1	40.5	65.1	55.7	59.9	41.4	46.5	42.0	41.6	50.5	30.8	28.7	42.7	49.4
SmolLM3-3B	38.9	55.2	29.4	34.4	31.4	40.6	30.1	47.3	47.3	45.5	46.3	23.1	31.7	45.6	26.4	45.7	47.8	35.9	45.2	33.9	49.6
Teuken-7B-instruct-v0.6	36.4	34.1	27.6	35.3	31.6	25.2	28.4	52.9	39.0	52.5	52.7	39.6	39.2	42.0	31.8	40.8	44.5	40.2	33.2	34.1	42.9
Open-Weight Models																					
gemma-3-12b-it	73.9	63.7	54.0	57.2	45.6	61.0	56.8	70.9	62.1	71.5	73.3	79.0	63.1	57.5	67.0	58.2	60.8	60.0	62.6	58.6	66.5
gemma-3-27b-it	79.8	69.9	60.8	63.9	51.0	66.3	64.0	72.9	63.9	82.4	78.4	74.5	65.0	66.0	72.8	64.2	68.5	59.1	66.2	64.9	71.4
Llama-3.1-8B-Instruct	63.1	54.8	40.5	46.5	36.2	39.9	45.1	62.0	57.2	64.5	62.4	52.9	45.8	49.1	51.0	52.8	46.8	51.7	53.6	46.8	61.8
Llama-3.3-70B-Instruct	82.8	69.7	64.3	64.0	53.2	62.0	65.0	76.1	76.0	81.2	82.6	75.1	69.0	69.1	75.6	65.6	69.2	62.1	70.6	83.2	74.6
Qwen2.5-72B-Instruct	79.9	72.2	55.0	64.0	42.8	59.0	65.8	74.7	86.5	83.5	85.0	60.6	68.0	68.9	63.6	65.3	69.0	59.9	69.1	64.1	75.0
Qwen3-32B	80.4	69.3	61.0	64.3	47.6	68.0	65.1	72.9	84.9	80.7	81.2	74.9	67.5	67.0	64.2	64.7	65.2	61.5	70.5	69.5	74.8
Qwen3-8B	64.7	59.1	49.0	53.0	39.4	56.0	54.7	67.5	73.9	73.3	70.9	51.1	58.8	62.5	54.2	57.8	58.5	53.0	60.6	59.0	67.3

Table 35: **INCLUDE by language Part 1 A-I**

	Languages																				
Model	**italian**	**japanese**	**kazakh**	**korean**	**lithuanian**	**malay**	**malayalam**	**persian**	**polish**	**portuguese**	**russian**	**serbian**	**spanish**	**tagalog**	**tamil**	**telugu**	**turkish**	**ukrainian**	**urdu**	**uzbek**	**vietnamese**
Fully Open Models																					
Apertus-70B-Instruct	70.9	77.3	54.2	61.6	67.0	62.8	34.0	45.5	55.1	59.0	59.4	69.2	71.1	75.8	40.8	37.2	57.5	67.6	25.3	47.8	63.1
Apertus-8B-Instruct	72.0	68.2	49.0	57.0	61.3	56.5	35.6	42.7	47.9	52.7	51.8	67.2	66.1	68.7	41.1	41.9	56.0	63.3	23.0	43.5	54.8
ALLaM-7B-Instruct-preview	61.3	52.5	33.4	48.2	42.2	47.9	30.9	37.0	32.6	50.6	46.6	47.4	63.1	61.8	33.0	27.8	38.9	55.7	33.4	32.9	41.3
EuroLLM-22B-Instruct-Preview	75.8	71.7	35.8	58.8	67.3	47.0	30.5	30.5	62.7	54.1	55.7	61.1	71.2	55.5	25.7	31.0	55.6	68.4	32.5	56.6	35.7
EuroLLM-9B-Instruct	66.7	64.2	34.4	58.6	64.0	39.2	32.8	36.7	52.7	55.3	53.1	58.6	72.9	55.2	27.5	25.7	50.1	62.9	19.2	35.4	32.0
K2-Chat	70.3	55.0	32.8	40.2	35.4	44.8	30.0	36.9	39.6	53.3	50.0	63.5	68.4	57.9	28.5	24.1	44.2	60.3	39.6	24.5	34.9
marin-8b-instruct	55.6	51.4	33.2	42.8	29.9	40.4	32.4	31.2	25.5	48.2	45.1	37.0	55.8	54.1	33.5	33.2	36.3	47.3	22.9	36.4	39.1
Minerva-7B-instruct-v1.0	41.7	29.2	28.2	26.4	26.2	25.6	27.5	24.5	10.7	29.7	21.2	26.6	27.4	22.6	25.2	23.9	26.3	33.6	23.6	21.9	27.1
OLMo-2-0325-32B-Instruct	72.4	66.9	36.2	49.4	49.5	64.5	40.3	42.3	44.4	57.7	52.8	64.2	71.0	75.5	44.8	37.1	50.1	60.5	23.1	36.2	52.8
OLMo-2-1124-7B-Instruct	51.5	39.5	29.4	31.0	36.0	41.5	22.5	32.0	29.0	42.5	40.6	29.7	42.5	59.2	31.0	29.6	34.6	42.1	28.8	29.8	37.6
salamandra-7b-instruct	65.2	45.7	36.4	39.6	50.1	37.7	31.7	29.7	43.0	42.8	47.8	65.0	59.6	46.3	23.5	23.8	36.1	60.9	18.0	20.2	35.1
SmolLM3-3B	69.3	58.3	33.6	55.4	31.7	38.0	21.2	37.5	38.5	50.8	48.0	43.3	64.6	39.5	27.7	25.4	31.7	45.9	23.2	21.6	43.5
Teuken-7B-instruct-v0.6	59.0	44.5	34.6	38.6	52.1	40.3	31.4	30.1	41.6	46.9	42.3	44.2	56.7	48.9	24.9	29.2	33.3	44.1	19.0	33.7	33.6
Open-Weight Models																					
gemma-3-12b-it	78.0	77.7	50.4	62.2	70.4	66.5	46.5	49.1	70.5	60.6	60.2	71.9	70.8	78.3	46.8	46.3	56.3	68.8	25.3	56.6	66.5
gemma-3-27b-it	79.3	83.2	54.4	66.6	78.9	71.2	48.8	54.2	78.4	65.0	63.3	80.5	82.2	83.8	56.9	57.2	61.8	69.2	32.9	66.6	66.1
Llama-3.1-8B-Instruct	74.9	63.7	47.0	55.6	50.8	58.6	30.4	41.7	58.2	53.6	58.7	61.3	67.4	62.5	34.9	35.9	60.7	59.6	30.4	51.2	63.7
Llama-3.3-70B-Instruct	93.2	83.7	61.6	65.2	82.5	74.5	49.4	58.3	89.4	74.4	68.1	83.3	82.3	81.9	57.8	52.2	71.2	74.5	56.1	64.8	77.5
Qwen2.5-72B-Instruct	87.9	89.0	51.8	68.8	80.0	71.0	50.3	55.1	79.3	68.9	67.1	84.3	82.9	80.4	42.1	50.1	62.0	76.8	58.9	60.5	74.5
Qwen3-32B	88.8	85.3	53.6	67.2	83.9	75.0	54.4	57.6	70.6	70.8	67.5	83.0	79.3	80.4	60.0	53.7	63.8	74.8	37.9	67.7	73.6
Qwen3-8B	82.5	77.6	43.4	63.0	65.2	62.9	46.4	52.5	58.3	63.3	58.5	73.6	73.9	68.1	42.9	40.4	55.6	70.4	30.0	50.0	63.7

Table 36: **INCLUDE by language Part 2 I-V**

	Languages																					
Model	**amharic**	**assamese**	**czech**	**dagbani**	**dangme**	**danish**	**darija**	**dogri**	**ekpeye**	**embu**	**esan**	**ewe**	**fante**	**fula**	**ga**	**gujarati**	**hausa**	**ibibio**	**idoma**	**igala**	**igbo**	**jju**	**kannada**
Fully Open Models																							
Apertus-70B-Instruct	41.4	34.9	55.2	29.4	28.0	85.0	69.2	43.7	36.0	41.2	39.5	29.1	25.8	44.8	31.0	41.7	31.6	33.1	43.6	38.6	39.2	30.4	38.4
Apertus-8B-Instruct	33.2	33.8	47.4	29.0	29.2	81.2	64.6	39.5	33.3	48.0	44.1	30.6	27.8	39.5	27.2	47.4	29.2	27.9	45.4	38.6	35.8	29.0	39.8
ALLaM-7B-Instruct-preview	31.2	29.0	40.0	26.0	28.2	68.4	75.2	36.1	34.7	40.1	38.2	25.7	25.2	41.0	27.0	34.3	29.8	37.1	34.0	32.7	29.2	29.2	29.2
EuroLLM-22B-Instruct-Preview	26.4	29.8	53.8	23.4	25.0	86.8	67.8	45.6	38.0	38.1	32.2	20.0	23.6	37.2	23.8	34.3	29.8	43.0	34.6	33.7	27.4	28.0	25.6
EuroLLM-9B-Instruct	27.0	27.9	47.8	23.4	26.0	83.2	65.4	42.6	37.0	36.7	42.1	24.3	30.0	38.7	28.0	27.7	31.2	39.7	43.8	37.3	33.4	34.4	31.4
K2-Chat	29.6	32.7	45.6	26.2	27.0	72.6	37.8	30.4	35.0	38.1	33.6	26.9	22.8	40.6	29.2	28.7	30.4	30.1	43.2	34.0	37.0	31.2	25.8
marin-8b-instruct	24.8	27.6	30.8	26.4	27.2	48.7	48.8	30.8	41.0	40.8	38.8	26.9	25.8	37.9	28.4	32.1	28.0	35.7	33.6	38.3	31.2	33.4	28.8
Minerva-7B-instruct-v1.0	24.0	26.8	27.6	23.6	26.4	39.0	34.4	24.7	30.0	28.6	38.2	29.4	24.2	32.6	28.4	25.9	22.4	32.0	36.6	31.7	26.4	30.0	27.8
OLMo-2-0325-32B-Instruct	36.8	37.5	46.0	31.4	26.6	80.0	57.2	39.5	36.7	47.6	38.8	26.3	28.8	42.5	31.6	44.9	27.6	26.5	35.4	38.6	37.0	29.4	39.2
OLMo-2-1124-7B-Instruct	31.8	31.2	26.4	27.0	25.4	49.3	47.8	32.3	33.0	37.8	32.2	27.1	28.2	39.8	27.0	32.4	27.6	27.6	34.8	32.7	31.4	29.0	29.8
salamandra-7b-instruct	30.4	22.4	39.4	25.4	25.2	68.8	38.8	27.4	31.3	32.3	35.5	26.6	26.6	29.9	20.2	27.7	26.8	32.7	37.2	31.4	32.6	32.6	26.2
SmolLM3-3B	31.2	30.9	33.2	25.0	22.8	49.3	49.6	33.5	27.7	38.8	32.2	26.9	28.6	32.6	24.4	30.5	22.8	22.1	28.6	31.4	35.8	23.4	26.4
Teuken-7B-instruct-v0.6	27.6	26.5	37.0	25.8	21.8	70.6	47.4	28.5	32.3	38.1	35.5	25.7	26.6	30.7	23.2	30.5	26.2	26.1	44.0	31.0	26.8	27.2	26.2
Open-Weight Models																							
gemma-3-12b-it	54.8	40.8	55.8	26.4	23.6	85.0	71.0	45.6	32.7	42.9	35.5	26.3	30.2	41.8	25.4	60.4	31.0	25.4	42.8	35.0	40.4	30.4	46.6
gemma-3-27b-it	64.4	46.0	62.0	27.2	27.0	92.2	77.2	50.2	39.7	45.2	46.7	24.6	27.0	46.0	25.8	66.4	37.4	36.8	41.0	38.9	39.0	34.4	56.2
Llama-3.1-8B-Instruct	25.2	35.3	48.8	24.6	22.8	71.3	55.2	34.2	23.7	39.5	28.9	21.1	29.6	37.2	23.4	42.4	27.2	26.8	32.0	30.4	35.8	26.8	29.4
Llama-3.3-70B-Instruct	49.2	48.9	67.6	31.6	27.6	94.4	70.4	54.4	35.0	49.0	38.2	29.7	30.0	47.1	27.8	66.7	34.8	29.8	36.6	41.3	40.4	31.6	55.8
Qwen2.5-72B-Instruct	41.8	54.4	67.0	26.8	23.8	88.1	72.6	47.5	37.0	46.3	27.6	24.9	28.8	44.1	20.2	67.3	25.4	27.2	40.6	37.6	34.4	23.4	49.6
Qwen3-32B	50.6	55.9	68.4	26.8	26.6	76.2	66.2	55.9	36.7	41.8	35.5	26.9	30.6	48.7	27.2	67.9	27.8	40.8	44.8	38.0	38.2	32.2	54.8
Qwen3-8B	43.8	46.7	53.6	27.4	22.6	62.6	61.6	41.8	33.0	46.9	32.2	24.9	25.6	37.9	26.8	53.0	27.0	25.0	39.6	36.3	33.2	29.8	43.2

Table 37: **INCLUDE V2 by language Part 1 A-K**

	Languages																					
Model	**kinyarwanda**	**luo**	**maithili**	**makhuwa**	**marathi**	**nyanja**	**obolo**	**oriya**	**oromo**	**punjabi**	**sena**	**sindhi**	**sinhala**	**slovak**	**somali**	**swahili**	**swedish**	**tangale**	**tigrinya**	**twi**	**tyap**	**yoruba**
Fully Open Models																						
Apertus-70B-Instruct	47.2	38.4	46.0	42.9	40.1	63.2	33.0	38.2	21.0	41.8	46.9	42.4	46.9	64.6	67.4	41.1	67.4	23.4	34.2	24.8	41.4	29.2
Apertus-8B-Instruct	38.6	37.2	56.3	33.5	42.4	53.4	31.8	36.8	25.0	40.2	37.1	46.5	40.0	61.5	41.7	34.2	58.4	25.3	29.6	26.4	39.8	26.0
ALLaM-7B-Instruct-preview	35.4	39.2	18.4	38.6	26.6	52.4	32.6	29.0	21.6	33.9	34.4	40.1	32.1	43.1	29.9	25.7	48.6	31.5	27.2	26.0	37.8	31.4
EuroLLM-22B-Instruct-Preview	29.4	31.4	47.9	33.9	34.5	45.2	30.0	30.9	25.6	31.3	41.2	35.0	26.2	65.4	35.2	28.3	64.4	36.3	26.0	28.2	38.0	26.8
EuroLLM-9B-Instruct	32.5	31.6	63.6	36.6	28.9	50.2	37.4	31.5	23.4	29.1	42.6	30.9	33.1	60.0	35.2	25.7	57.6	26.4	29.8	26.2	37.0	27.8
K2-Chat	34.1	35.6	19.9	35.8	28.0	48.2	30.0	28.8	24.0	29.9	29.5	25.5	30.3	56.9	46.6	26.7	55.2	35.2	26.4	25.6	32.2	26.8
marin-8b-instruct	40.7	34.6	46.0	43.3	26.3	50.4	34.6	32.5	23.8	33.9	34.6	32.8	32.8	36.9	59.1	27.3	42.0	27.1	28.6	29.8	37.8	28.4
Minerva-7B-instruct-v1.0	28.9	26.0	50.2	26.0	24.0	28.8	25.0	30.4	23.6	26.7	15.4	28.3	33.4	27.7	15.5	27.1	23.0	23.4	27.8	23.0	29.0	25.6
OLMo-2-0325-32B-Instruct	37.5	42.8	9.6	42.9	37.8	57.4	28.0	39.0	27.8	45.4	43.4	39.2	47.6	60.0	63.3	33.8	66.8	30.8	34.4	29.8	41.2	32.4
OLMo-2-1124-7B-Instruct	30.4	35.2	3.4	40.9	30.6	48.4	32.4	29.8	25.2	29.5	32.8	27.1	31.7	31.5	29.9	27.5	38.0	29.3	26.0	25.2	34.6	26.6
salamandra-7b-instruct	29.7	28.0	80.8	25.6	27.0	38.0	32.6	26.3	25.6	29.9	31.1	25.2	27.2	48.5	53.4	27.5	48.8	30.4	33.8	25.8	34.0	26.4
SmolLM3-3B	28.9	32.6	16.1	36.6	29.6	46.4	27.6	29.0	28.8	30.1	23.0	34.4	24.5	43.8	31.4	29.5	45.6	37.4	26.2	26.6	34.6	27.8
Teuken-7B-instruct-v0.6	31.2	32.8	14.9	42.5	23.4	44.0	32.8	26.9	24.8	23.9	36.1	26.4	29.7	40.8	29.5	29.3	47.4	30.8	24.0	24.8	33.8	24.6
Open-Weight Models																						
gemma-3-12b-it	44.9	33.6	33.0	39.0	50.3	58.4	29.2	45.4	24.8	55.6	34.4	50.3	52.8	78.5	71.2	42.6	74.4	25.6	38.6	27.8	38.4	27.8
gemma-3-27b-it	45.9	36.4	34.9	43.7	55.6	66.6	34.2	50.8	24.6	61.2	50.2	57.6	64.1	86.9	45.1	45.4	82.8	27.5	44.8	27.2	44.6	29.2
Llama-3.1-8B-Instruct	37.5	30.0	18.8	33.9	36.2	48.4	22.0	33.3	22.2	38.0	31.8	41.1	43.8	47.7	38.3	33.0	62.4	30.4	20.2	27.0	37.0	27.2
Llama-3.3-70B-Instruct	43.3	39.4	11.9	43.7	55.9	64.6	34.0	47.6	26.2	61.8	39.1	59.9	65.5	90.0	37.5	45.2	83.8	26.4	34.8	30.8	47.6	33.4
Qwen2.5-72B-Instruct	38.3	34.4	11.1	35.0	54.6	60.0	32.2	45.7	24.0	60.4	36.1	49.7	53.8	93.8	45.8	32.6	84.0	24.2	32.4	25.8	41.4	28.8
Qwen3-32B	34.6	38.8	39.1	46.5	66.8	54.8	34.4	51.3	28.0	63.1	39.3	55.1	51.4	90.8	58.3	33.8	83.4	32.6	39.6	29.6	40.8	30.0
Qwen3-8B	34.4	37.4	10.0	40.6	55.6	49.2	30.4	45.2	26.8	49.6	25.4	48.1	42.4	77.7	32.2	34.0	73.4	34.8	28.8	25.4	39.0	28.8

Table 38: **INCLUDE V2 by language Part 2 K-Y**

	Languages										
Model	**bn**	**de**	**en**	**es**	**fr**	**ja**	**ru**	**sw**	**te**	**th**	**zh**
Fully Open Models											
Apertus-70B-Instruct	58.8	69.6	72.0	71.2	64.4	62.4	76.0	49.6	58.4	60.4	64.8
Apertus-8B-Instruct	44.0	52.8	64.0	57.6	51.6	44.4	54.4	26.0	38.8	49.6	50.0
ALLaM-7B-Instruct-preview	3.2	43.6	63.2	42.8	42.0	21.6	35.6	8.0	3.6	11.6	44.8
EuroLLM-22B-Instruct-Preview	27.2	73.2	76.8	71.2	68.4	60.8	72.0	8.8	8.4	22.4	68.4
EuroLLM-9B-Instruct	0.8	61.2	63.6	57.2	50.8	41.2	58.8	2.0	1.2	8.8	51.2
K2-Chat	12.8	73.2	83.2	70.8	71.2	53.2	72.4	10.8	2.8	14.8	74.8
marin-8b-instruct	6.8	51.2	75.6	62.4	59.6	48.4	63.2	2.0	12.4	37.6	51.2
Minerva-7B-instruct-v1.0	1.2	2.0	9.2	6.8	3.6	0.4	2.4	1.2	0.4	1.6	1.6
OLMo-2-0325-32B-Instruct	63.6	74.8	86.8	78.4	75.6	67.6	72.8	40.8	49.2	57.6	73.6
OLMo-2-1124-7B-Instruct	12.4	49.6	73.6	61.2	55.6	23.2	53.6	6.4	8.0	15.2	47.6
salamandra-7b-instruct	0.4	16.4	22.4	16.0	17.6	2.8	16.4	2.4	0.0	0.0	10.8
SmolLM3-3B	1.6	67.2	74.4	65.6	61.2	56.0	65.2	12.0	0.8	34.8	58.8
Teuken-7B-instruct-v0.6	2.0	37.2	39.6	36.8	32.4	5.6	27.6	3.2	2.0	2.4	22.0
Open-Weight Models											
gemma-3-12b-it	48.4	80.0	83.6	82.8	72.8	75.6	79.2	69.2	76.0	77.6	13.2
gemma-3-27b-it	34.4	85.2	83.2	81.2	74.8	80.8	78.8	76.4	80.8	82.8	30.8
Llama-3.1-8B-Instruct	63.6	70.0	78.8	74.4	71.2	63.6	71.6	59.2	55.6	72.0	64.4
Llama-3.3-70B-Instruct	87.6	88.4	84.8	88.0	87.6	86.8	84.8	82.8	81.2	88.4	85.6
Qwen2.5-72B-Instruct	84.0	81.2	85.6	80.8	77.2	80.8	80.8	56.0	65.6	70.4	75.6
Qwen3-32B	79.2	83.2	86.8	82.0	76.4	81.6	82.0	46.8	75.2	64.8	56.0
Qwen3-8B	59.6	16.0	85.6	22.8	65.6	70.4	79.6	34.4	50.0	30.0	58.0

Table 39: **MGSM by language**

	Languages				
Model	**de**	**en**	**fr**	**it**	**rm**
Fully Open Models					
Apertus-70B-Instruct	69.4	68.1	68.5	67.5	62.5
Apertus-8B-Instruct	64.0	64.5	63.8	63.5	59.5
ALLaM-7B-Instruct-preview	58.1	62.0	59.1	55.8	45.0
EuroLLM-22B-Instruct-Preview	64.1	65.8	64.7	64.2	49.1
EuroLLM-9B-Instruct	64.2	64.8	63.4	62.3	51.2
K2-Chat	64.1	66.0	64.6	63.5	51.8
marin-8b-instruct	55.3	59.6	54.8	52.3	46.1
Minerva-7B-instruct-v1.0	35.9	41.2	36.9	38.7	34.5
OLMo-2-0325-32B-Instruct	65.3	66.9	64.2	61.9	52.6
OLMo-2-1124-7B-Instruct	47.2	55.5	49.7	44.6	39.1
salamandra-7b-instruct	62.4	62.1	61.7	60.6	51.2
SmolLM3-3B	58.9	61.5	59.3	57.6	45.8
Teuken-7B-instruct-v0.6	55.9	56.7	56.5	53.6	42.4
Open-Weight Models					
gemma-3-12b-it	67.8	67.8	67.5	66.6	56.0
gemma-3-27b-it	73.2	73.5	72.6	72.1	63.6
Llama-3.1-8B-Instruct	62.9	65.1	62.9	61.1	48.1
Llama-3.3-70B-Instruct	76.3	76.3	75.7	75.0	68.1
Qwen2.5-72B-Instruct	72.8	72.9	71.9	71.2	61.2
Qwen3-32B	68.7	69.3	68.2	66.4	55.4
Qwen3-8B	62.8	64.9	62.4	61.1	52.2

Table 40: **Switzerland QA by language**

	Languages																													
Model	**ar**	**bn**	**ca**	**da**	**de**	**es**	**eu**	**fr**	**gu**	**hi**	**hr**	**hu**	**hy**	**id**	**it**	**kn**	**ml**	**mr**	**ne**	**nl**	**pt**	**ro**	**ru**	**sk**	**sr**	**sv**	**ta**	**te**	**uk**	**vi**	**zh**
Fully Open Models																															
Apertus-70B-Instruct	54.1	51.4	54.6	56.9	56.9	56.3	49.6	57.4	48.1	50.7	55.8	54.4	46.4	56.9	56.0	49.5	49.9	50.0	49.9	56.9	55.3	54.2	56.4	52.6	57.1	57.6	49.3	48.2	55.3	57.4	55.2
Apertus-8B-Instruct	51.6	51.7	53.7	52.5	54.1	54.8	49.1	54.8	47.0	48.7	53.0	51.2	43.8	53.6	55.4	51.2	48.5	48.8	48.3	54.7	55.7	55.1	57.1	51.6	55.0	55.8	48.9	49.2	53.4	55.2	55.2
ALLaM-7B-Instruct-preview	46.1	43.0	44.4	41.0	45.0	42.0	41.3	44.2	42.9	40.7	42.7	42.5	46.9	40.3	45.2	45.2	47.5	45.5	42.7	43.2	41.3	40.0	46.3	41.8	41.8	44.5	48.2	46.3	44.4	44.6	44.9
EuroLLM-22B-Instruct-Preview	49.4	46.6	50.5	51.9	53.5	52.9	41.2	52.4	46.3	47.9	52.5	51.9	45.3	46.6	51.4	46.0	45.7	46.1	46.6	53.9	52.8	52.9	54.0	48.9	54.0	51.9	50.5	45.9	50.5	49.8	51.3
EuroLLM-9B-Instruct	48.5	47.8	47.7	47.3	51.5	47.9	40.6	46.0	39.5	46.1	50.3	50.8	43.8	44.3	46.2	40.4	47.5	44.1	38.1	49.4	45.8	47.0	47.7	45.7	52.7	46.3	49.7	47.6	47.1	45.6	47.7
K2-Chat	50.5	51.6	50.1	50.3	50.6	51.1	41.3	52.1	41.9	47.3	53.1	48.3	44.1	50.3	51.6	42.2	48.6	44.9	45.2	49.5	50.1	51.4	50.3	45.2	54.3	51.3	49.4	48.8	51.8	50.5	53.0
marin-8b-instruct	48.6	47.9	45.9	46.9	52.5	52.8	42.0	52.8	43.7	48.6	45.8	46.5	44.1	47.0	49.2	44.4	46.2	46.4	44.8	47.6	49.7	47.2	53.1	41.5	47.1	46.1	50.0	47.4	47.9	48.5	52.2
Minerva-7B-instruct-v1.0	49.0	50.3	39.9	44.7	44.6	45.3	43.2	48.1	46.7	48.9	50.6	51.5	46.5	43.7	46.0	49.2	49.9	49.5	47.3	42.9	46.0	47.1	49.2	42.6	50.7	46.0	51.1	49.1	47.4	48.5	48.3
OLMo-2-0325-32B-Instruct	52.6	50.1	58.1	60.4	65.2	65.8	41.3	60.9	47.0	55.0	59.8	55.8	42.8	62.5	62.4	47.7	47.2	47.9	52.2	63.6	64.1	61.0	64.1	54.1	59.5	64.4	51.6	47.4	56.2	58.9	60.5
OLMo-2-1124-7B-Instruct	45.6	48.5	46.1	44.0	51.0	48.9	42.8	50.6	44.8	44.1	43.2	45.6	43.0	48.5	48.7	44.6	44.3	47.6	44.4	45.8	50.0	45.5	53.1	38.4	44.5	46.6	49.1	45.1	45.9	48.7	49.6
salamandra-7b-instruct	50.3	49.5	48.8	48.4	50.5	48.5	45.3	49.8	45.9	49.1	52.1	49.0	42.8	43.9	47.8	46.6	47.8	48.6	47.0	49.0	49.1	46.9	49.1	48.7	50.6	48.3	49.1	47.2	52.6	48.8	47.8
SmolLM3-3B	52.8	51.4	48.6	46.4	55.4	52.6	43.7	54.7	45.1	48.3	51.3	50.7	44.9	50.0	52.3	46.9	49.7	48.4	47.0	48.7	54.3	51.0	54.8	45.4	52.0	47.7	50.9	48.0	49.4	52.3	51.1
Teuken-7B-instruct-v0.6	51.2	48.1	49.5	44.5	43.5	49.8	43.1	49.1	48.5	50.4	52.9	48.0	45.7	42.2	49.3	49.3	51.6	49.3	49.2	46.6	44.3	47.8	50.9	48.0	50.1	46.6	51.8	49.7	49.5	42.5	47.6
Open-Weight Models																															
gemma-3-12b-it	57.6	52.8	56.5	59.9	58.1	57.6	48.4	59.0	48.5	53.0	60.3	58.0	48.3	60.0	60.2	51.0	51.1	50.7	51.9	59.3	60.4	59.7	58.3	54.8	59.4	58.7	53.0	50.1	59.7	60.0	56.1
gemma-3-27b-it	55.9	52.9	54.5	60.0	59.9	55.6	46.4	59.3	48.5	52.8	56.5	53.0	46.6	58.5	57.7	50.9	49.7	50.1	51.5	57.9	58.2	58.4	58.3	55.9	57.0	59.1	49.0	48.0	59.1	56.4	56.0
Llama-3.1-8B-Instruct	51.6	49.9	50.5	53.1	53.5	54.3	44.7	55.6	44.6	49.0	52.6	50.4	44.1	52.6	52.8	47.1	46.3	49.3	47.4	54.4	53.2	51.5	55.8	49.5	53.0	53.5	48.0	46.1	52.4	52.3	50.6
Llama-3.3-70B-Instruct	53.4	53.9	57.4	60.3	63.2	60.1	48.6	62.1	49.2	54.1	54.4	53.8	44.9	57.3	61.6	48.1	47.3	51.1	51.2	61.3	59.1	58.8	60.6	52.0	53.4	60.9	49.7	47.3	53.8	58.2	58.3
Qwen2.5-72B-Instruct	59.1	57.0	58.8	62.5	63.6	62.5	47.8	65.9	52.0	58.6	58.4	56.1	44.6	63.1	63.7	50.4	53.4	54.6	53.5	64.6	64.3	58.4	63.4	58.3	61.7	62.9	50.8	52.3	59.9	63.1	63.0
Qwen3-32B	48.6	51.6	51.6	53.6	52.0	51.5	49.3	53.7	44.2	52.4	53.2	52.2	43.4	51.2	53.0	47.2	50.7	48.8	48.2	52.2	50.8	47.4	51.4	50.9	53.6	54.2	49.0	46.6	51.1	53.9	51.8
Qwen3-8B	49.0	53.2	48.0	52.4	53.7	49.6	50.1	53.1	47.1	52.7	54.9	51.0	47.1	54.0	49.6	52.1	48.2	50.9	51.2	52.1	49.8	50.1	52.6	51.7	55.8	53.7	52.9	50.5	52.1	50.6	52.1

Table 41: **TruthfulQA Multilingual by language**

	Context Length				
Model	4k	8k	16k	32k	64k
Apertus-8B	89.5	82.1	75.8	70.3	55.9
Apertus-70B	88.3	80.2	77.7	71.1	56.9
Apertus-8B-Instruct	91.2	87.2	79.1	65.9	61.4
Apertus-70B-Instruct	94.8	89.9	85.7	81.9	67.3
Qwen3-8B	94.2	93.1	91.6	89.7	75.7
Qwen3-32B	94.4	94.1	93.5	92.6	87.1
Qwen2.5-72B-Instruct	96.1	95.0	94.5	93.3	89.3
Llama-3.1-8B	93.1	91.5	90.4	85.7	81.3
Llama-3.1-8B-Instruct	95.0	94.0	91.8	86.2	84.8
Llama-3.3-70B-Instruct	95.2	94.7	94.8	93.7	85.0
Gemma-3-12b-it	89.6	84.6	77.5	72.1	61.0
Gemma-3-27b-it	92.7	85.4	79.8	68.7	58.0
SmolLM3-3B	88.4	83.9	81.8	76.6	65.9

Table 42: **Results on RULER Benchmark Across Various Context Lengths**. Evaluation of Apertus-70B-Instruct on 64k context length exceeded the maximum allowed runtime on the node.

	Rumantsch Grischun		Sursilvan		Sutsilvan	
Model	DE to RM ↑	RM to DE ↑	DE to RM ↑	RM to DE ↑	DE to RM ↑	RM to DE ↑
Apertus-8B-Instruct	22.9	41.0	12.8	31.1	6.8	20.9
Apertus-70B-Instruct	28.0	44.7	15.1	36.9	7.6	27.2
Llama-3.3-70B-Instruct	21.6	35.6	11.9	28.0	6.2	17.8
	Surmiran		**Puter**		**Vallader**	
Model	DE to RM ↑	RM to DE ↑	DE to RM ↑	RM to DE ↑	DE to RM ↑	RM to DE ↑
Apertus-8B-Instruct	9.1	26.8	5.7	27.2	10.1	31.4
Apertus-70B-Instruct	10.8	33.7	9.9	32.7	12.3	38.4
Llama-3.3-70B-Instruct	7.9	22.1	8.7	27.5	11.0	31.6

Table 43: **Post-training Evaluation**: BLEU scores for machine translation between German and six varieties of Romansh, based on the Romansh WMT24++ benchmark. Higher scores are better.

Exposure	Rouge-L (↓)		LCCS (↓)		TTR (↑)		
Frequency	greedy	nucleus	greedy	nucleus	greedy	nucleus	ground truth
0	0.178	0.175	0.010	0.010	0.229	0.500	0.538
1	0.184	0.178	0.011	0.010	0.220	0.496	0.535
2	0.183	0.179	0.010	0.009	0.219	0.497	0.539
4	0.182	0.175	0.010	0.010	0.221	0.499	0.538
8	0.183	0.175	0.010	0.009	0.230	0.500	0.538
16	0.184	0.177	0.010	0.010	0.235	0.499	0.537
32	0.185	0.180	0.011	0.010	0.246	0.499	0.536
64	0.184	0.179	0.011	0.010	0.270	0.503	0.539
128	0.188	0.180	0.013	0.012	0.313	0.504	0.540

Table 44: **Impact of Decoding Strategy on Memorization and Text Degeneration for Apertus 70B.** Metrics are averaged across Gutenberg V1 and V2 at a fixed offset of 0, computed on 500-token suffixes conditioned on 500-token prefixes. The table compares greedy and nucleus sampling across exposure frequencies. Under greedy decoding, significant degeneration occurs, yet TTR still increases moderately from ∼0.225 for unseen sequences to 0.313 at the highest exposure (a gain of 44 unique tokens). In contrast, nucleus sampling maintains high lexical diversity (TTR ≈0.500). Crucially, verbatim recall metrics (Rouge-L, LCCS) remain at baseline for both strategies, confirming that our applied mitigation is robust and not an artifact of text degeneration. The arrows (↑,↓) show the desired direction for each metric.

	Bias	Toxicity & Safety				
Model	BBQ (↑)	ToxiGen (↑)	HarmBench (↓)	HarmBench Direct Request (↓)	HarmBench Human Jailbreaks (↓)	RealToxicityPrompts LlaMA-Guard3 Subsampled (↓)
Fully Open Models						
Apertus-70B-Instruct	67.4	70.3	31.9	10.3	36.2	0.2
Apertus-8B-Instruct	63.9	80.2	35.2	16.2	39.0	0.2
ALLaM-7B-Instruct-preview	57.7	84.3	7.0	2.8	7.9	1.6
EuroLLM-22B-Instruct-Preview	66.3	82.3	8.0	5.3	8.5	0.2
EuroLLM-9B-Instruct	65.0	51.5	6.0	3.4	6.6	0.0
K2-Chat	68.4	83.2	24.1	15.3	25.9	1.0
marin-8b-instruct	70.7	66.0	5.1	5.6	5.0	0.1
Minerva-7B-instruct-v1.0	45.7	50.7	33.9	23.8	35.9	1.3
OLMo-2-0325-32B-Instruct	76.6	78.0	22.5	9.7	25.1	0.4
OLMo-2-1124-7B-Instruct	63.8	85.1	10.7	4.1	12.0	0.4
salamandra-7b-instruct	63.9	81.3	14.5	10.3	15.4	4.2
SmolLM3-3B	69.5	56.7	51.1	50.6	51.2	1.7
Teuken-7B-instruct-v0.6	57.9	56.8	45.3	53.3	43.7	0.5
Open-Weight Models						
gemma-3-12b-it	75.2	86.7	42.2	25.0	45.7	0.3
gemma-3-27b-it	74.5	86.3	49.4	29.1	53.5	0.1
Llama-3.1-8B-Instruct	73.6	84.7	38.1	18.8	42.0	0.4
Llama-3.3-70B-Instruct	72.0	87.4	38.8	24.7	41.6	0.5
Qwen2.5-72B-Instruct	70.8	86.2	10.6	13.1	10.1	0.0
Qwen3-32B	78.4	85.9	12.0	11.6	12.1	0.1
Qwen3-8B	72.9	84.0	16.2	10.3	17.4	0.2

Table 45: **Post-training Evaluation:** Performance of Apertus models on benchmarks assessing **safety** and **security**. The arrows (↑,↓) show the desired direction for each benchmark.

Harm Category	ar	bn	cs	en	hu	ko	ms	ru	sr	th	vi	zh
Crimes & Illegal	41.14	40.83	39.84	39.09	40.28	43.99	40.21	39.76	39.16	39.39	38.14	39.66
Explicit Content	48.67	49.33	48.20	49.56	48.93	47.91	50.39	48.06	45.04	51.70	49.56	47.76
Fairness & Justice	56.30	50.00	55.95	57.76	55.99	51.86	54.54	56.87	54.58	56.07	57.21	56.45
Harm & Misuse	40.64	41.86	42.37	42.01	40.78	41.17	41.83	41.80	41.81	42.27	41.66	42.33
Privacy & Property	49.29	50.77	52.60	55.42	57.07	51.98	54.06	51.59	52.82	54.94	51.18	52.35

Table 46: **Severity-weighted scores for Apertus-70B-Instruct for each harm category across 12 languages**. Lower scores indicate better performance at detecting and handling harmful content.

Harm Category	ar	bn	cs	en	hu	ko	ms	ru	sr	th	vi	zh
Crimes & Illegal	44.64	46.10	45.50	42.46	47.26	47.29	47.41	44.18	46.06	44.09	42.80	43.11
Explicit Content	49.58	54.79	51.83	51.11	54.62	50.42	52.99	48.14	49.18	54.81	53.44	51.25
Fairness & Justice	59.05	59.83	61.46	59.09	61.96	59.88	62.64	59.53	63.98	59.49	61.72	59.91
Harm & Misuse	41.57	42.39	44.65	43.99	43.46	42.19	44.80	41.98	45.58	43.13	43.32	40.94
Privacy & Property	52.48	55.32	59.25	58.31	58.05	55.43	55.26	54.86	60.53	53.85	55.52	51.77

Table 47: **Severity-weighted scores for Apertus-8B-Instruct for each harm category across 12 languages.** Lower scores indicate better performance at detecting and handling harmful content.

G Infrastructure, Scaling, and Efficiency

The training of the Apertus collection of models was enabled by a leading supercomputing infrastructure. In the following, we detail the architectural features of the Machine Learning Platform and the engineering contributions that facilitated this release.

G.1 Infrastructure

G.1.1 The Research Infrastructure

The Research Infrastructure is an HPE Cray EX system with a measured HPL performance of 434 PFlops (fp64), placing it in the top 10 most powerful supercomputers globally.

Architecturally, the research infrastructure is designed so that resources operate as independent endpoints within a global high-speed network. This design addresses the limitations of traditional, vertically integrated HPC architectures by providing greater flexibility and composability.

The hardware infrastructure features 10,752 NVIDIA Grace-Hopper (GH200) GPUs (four per node), interconnected by a Slingshot-11 network with 200Gb/s injection bandwidth per GPU. For storage, Alps includes a 100PB ClusterStor HDD system and a 3PB ClusterStor SSD system, both using the Lustre file system, in addition to a 1PB VAST storage system. Additional details are outlined in Martinasso et al. (2025); Schuppli et al. (2025).

The research infrastructure uses a versatile software-defined cluster (vCluster) technology, which bridges cloud and HPC paradigms. This technology abstracts infrastructure, service management, and user environments into distinct layers, facilitating the deployment of independent, tenant-specific, and platform-specific services.

G.1.2 The Machine Learning Platform

The Machine Learning (ML) platform within the Research Infrastructure is specifically designed to meet the evolving computational demands of Artificial Intelligence (AI) and Machine Learning workloads. During the Apertus training runs, this platform leveraged a dedicated vCluster with approximately 1,500 NVIDIA Grace-Hopper (GH200) nodes (with 4 GPUs each) of the Alps system. This vCluster, named Clariden, ensures robust performance and scalability for training advanced AI models, including large language models (LLMs), and supports long-duration jobs. It is explicitly engineered to diverge from traditional High-Performance Computing (HPC) offerings, addressing specific challenges observed since its early access phase in 2023 (Schuppli et al., 2025).

A container-first approach is fundamental to the ML platform's design, streamlining the user experience, and enhancing application portability. Recognizing that ML users are typically familiar with container-based workflows and vendor-curated images (*e.g.*, PyTorch, JAX), the platform provides an environment that closely mirrors their existing setups, minimizing the need for extensive HPC-specific knowledge. This is facilitated by the Container Engine (CE) toolset, which runs Linux application containers on HPC resources in a seamless manner, incorporating Open Container Initiative (OCI) hooks and Container Device Interface (CDI) specifications for performance optimization. Users define their software environments concisely using TOML-based Environment Definition Files (EDF), promoting autonomy and rapid integration of custom dependencies crucial for the fast-evolving ML field (Cruz and Madonna, 2024).

To enhance the reliability and efficiency of large-scale ML training, the platform incorporates a node-vetting and early-abort system. This system dynamically verifies the readiness of the allocated compute nodes through lightweight, rapid diagnostic tests just prior to job execution. These tests are designed to identify transient issues such as high GPU temperature, insufficient memory, "dirty" GPU states, or network congestion that could otherwise degrade performance or cause job failures. The results are centrally collected, providing shared operational intelligence to improve the overall reliability of the system.

The pretraining and finetuning workloads of the Apertus models represent the first and most significant computational workload executed so far on the Research Infrastructure, running, for the 70B model, at scales from 2048 to 4096 GPUs over several months. The vCluster technology brought an operational flexibility unusual in HPC systems: critical updates could be applied selectively to vClusters serving other communities while being deferred on the nodes dedicated to this campaign, and the ML engineering team itself could roll out node-level changes without depending on system engineering colleagues. Crucially, this work demonstrated that even amid these technological advancements, delivered stable, well-scaling performance for cutting-edge large models pretraining.

G.2 Full Training Run Performance

A detailed timeline showing token throughput performance over the pretraining runs of the 70B and 8B Apertus models is displayed in Figure 13. We estimate that training of the 70B model for 15T tokens took 6.74×10^{24} FLOPs (details in Appendix G.4). In terms of usage, the main run has consumed around 6 million GPU hours, though this is underestimated as it does not count loading weights or overhead due to initial performance short-comings, failures or downtime. Once a production environment has been set up, we estimate that the model can be realistically trained in approximately 90 days on 4096 GPUs, accounting for overheads. If we assume 560 W power usage per Grace-Hopper module in this period, below the set power limit of 660 W, we can estimate 5 GWh power usage for the compute of the pretraining run. The infrastructure is almost carbon neutral, relying entirely on hydropower, and uses a sustainable cooling system

G.3 Engineering Challenges and Solutions

Training the Apertus model required careful, coordinated engineering across the entire software stack and a close collaboration with researchers. Engineers systematically tuned of software, hardware, and operational layers, to optimize a stable and highly-performant training pipeline capable of sustaining large-scale LLM training on 1024 nodes (4096 GPUs) with predictable convergence behavior. The following sections describe the key areas where improvements were made and the impact is illustrated in Figure 14.

G.3.1 Systems-level Fixes

The system relies on the HPE Slingshot 11 interconnect to provide the bandwidth required for large-scale distributed training. To enable efficient communication over this fabric, NCCL operates through the AWS OFI NCCL plugin in conjunction with libfabric. In the early stages, we observed significant performance variability caused by mismatched versions of these components. Aligning the plugin and libfabric versions resolved these inconsistencies, restoring stable communication and eliminating slowdowns during checkpoint restarts.

We resolved several other critical issues in collaboration with industry partners. One problem originated in the GPU driver, where an access-counter-based page migration bug caused interrupt storms on certain CPU cores, resulting in unpredictable performance when application threads were scheduled on those cores. As a workaround, we disabled the feature and eliminated this behavior. A second issue involved a race condition in the Linux kernel that could be triggered by GPU driver calls, leading to kernel panics and node crashes. A targeted Linux kernel hot patch corrected this problem and substantially improved system stability. Furthermore, we found that transparent huge pages in the Linux kernel negatively affected performance for this workload. To mitigate this, we introduced a Slurm option that allowed users to disable transparent huge pages when necessary.

Another challenge was the GH200 system's unified memory architecture, which combines two different types of memory: LPDDR5 for the CPU and HBM for the GPU. The Linux kernel and various system processes were not designed for this level of heterogeneity and sometimes allocated GPU memory, causing issues for applications that expected exclusive control over it. We mitigated this issue by explicitly binding many system processes and adding extra parameters to kernel calls. For example, we limited the memory implicitly allocated by the kernel in tmpfs filesystems, which are not directly constrained by user-space cgroups. These memory issues were compounded with another problem that resulted in OS file caches not migrating automatically back to CPU memory. This issue can only be fully addressed by a driver update; as an interim solution, the file caches are explicitly flushed and a Slurm prolog script verifies at least 90% of GPU memory can be allocated before a compute node is added to an allocation.

Together, these fixes removed major sources of instability and allowed large jobs to run for their full allocation without interruption.

G.3.2 Stability and Container Robustness

Ensuring the stability of the software environment was a major focus of our efforts. One issue stemmed from Triton's JIT kernel caches, which were originally stored on the distributed filesystem. This design introduced contention and, in some cases, race conditions across nodes that led to software crashes. By moving these caches to in-memory storage on each node, we eliminated race conditions and overall stability improved.

Container-level library compatibility posed another challenge. Early training runs were based on NGC 25.01, which contained a libnvrtc bug that caused sporadic crashes. The fix was present in

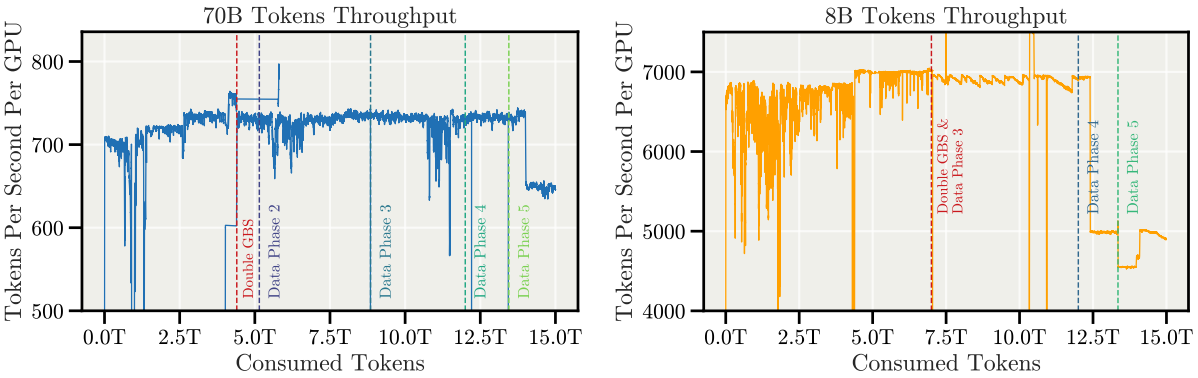

Figure 13: **Token Throughput During Training.** Left panel: 70B parameter model, Right panel: 8B model

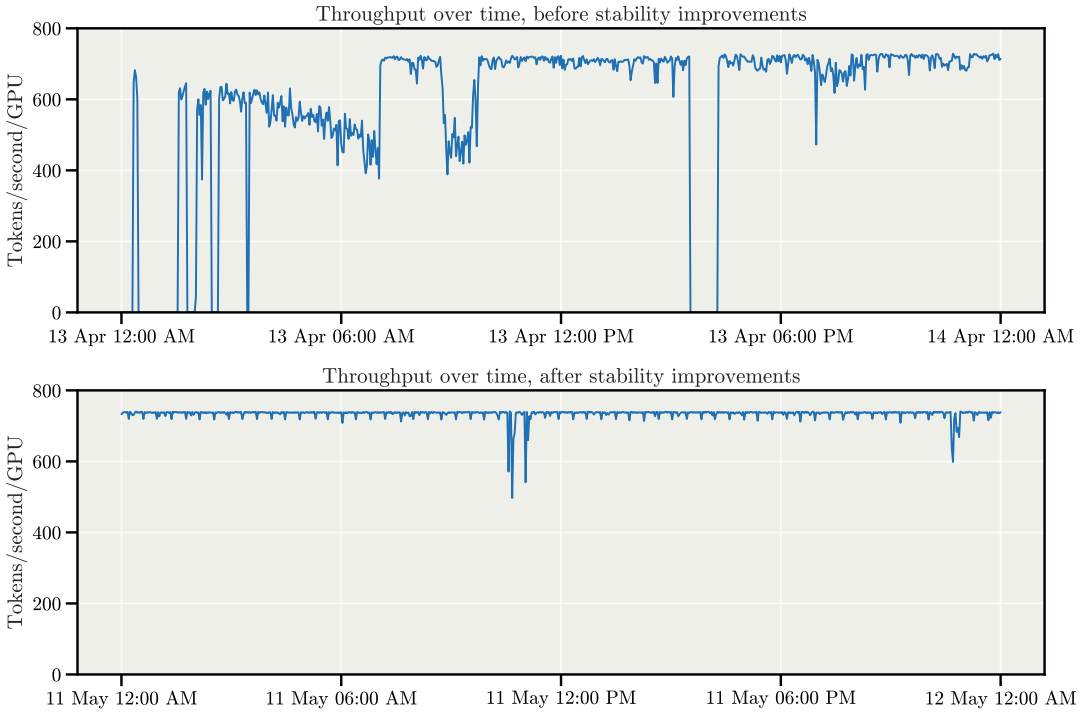

Figure 14: **Throughput of the 70B Apertus Pretraining on 2048 GPUs Before and after Stability Improvements. Top:** Runs prior to stability tuning show high variability, largely driven by poor filesystem I/O before migrating to full-flash storage, and an NVIDIA driver issue related to access counter-based memory page migration. **Bottom:** Performance after stability enhancements, exhibiting consistent throughput with predictable dips corresponding to Python garbage collection and checkpointing. Residual irregular fluctuations are attributable to minor filesystem interference.

later container releases, but dependencies on a specific PyTorch version in NGC 25.01 prevented an immediate upgrade. To address this, we built a custom container that included an updated version of libnvrtc, and once dependencies stabilized, it was possible to transition to NGC 25.03.

G.3.3 Checkpointing and Restart Strategies

Checkpointing is critical for fault tolerance, especially when operating at scale. We optimized storage usage by placing datasets and caches on high-IOPS SSD storage, which accommodate small, random reads; we stored checkpoint files, which involve large sequential writes, on high-capacity HDDs. The frequency of checkpointing (every 250 iterations) was determined using the Young/Daly formula, balancing checkpoint overhead (a few seconds) against expected mean time between failures (MTBF, a few hours) to minimize lost work and downtime. These strategies reduced the cost of restarts and ensured that long training runs could progress reliably even in the event of node failures.

To ensure continuous execution of the training process, each job submitted the next job to the queue once a basic initialization check completed successfully, indicating that the job would proceed seamlessly. We leveraged the Slurm sbatch configuration flag –dependency=singleton, which enforces that only one job with the same name and user can run simultaneously. To avoid wasting compute resources, we also employed the –signal option to send a SIGUSR2 signal a few minutes before the job's time limit, ensuring sufficient time to store a checkpoint and terminate gracefully.

G.3.4 Performance Optimizations at Scale

Beyond stability and resilience, we introduced targeted performance optimizations to maximize efficiency. One such improvement was to enable NVIDIA's vBoost feature through a custom Slurm option. This adjustment trades-off chip memory power to give it to the cores thus increasing GPU clock frequencies while remaining within power budgets. LLM workloads benefit from this as they are typically compute-bound, not memory-bound. We also identified periods during training that involved numerous small collective operations. By adjusting Megatron's distributed data parallel bucket size, many small NCCL collectives were consolidated into fewer, larger messages. This change significantly reduced communication latency and improved training performance during communication-heavy phases. Scaling to 1024 nodes was made possible with two key modifications to the model parallelism: first, removing delayed computation of the embedding gradients that caused spurious training metrics late in pretraining, assumed to be a Megatron issue, and second, increasing virtual pipelining within model-parallel groups to optimize communication patterns. Finally, to speed-up loading the container image, nearly 20GB in size, effectively at scale, Lustre striping had to be properly set for these files.

G.3.5 Operational Efficiency and Monitoring

Improving operational resilience was essential for reducing downtime and maximizing system utilization. We created a dedicated Slurm partition for large-scale jobs, ensuring resource availability for restarts and minimizing scheduling delays. Additional nodes were allocated to these partitions so that, in the event of hardware failure, spare capacity was immediately available. The queue time limits were extended to 48 hours to accommodate large jobs that required longer execution windows. In addition to these changes, we minimized downtime with automated exit triggers, signal handling, and continuous monitoring tools to detect and respond to anomalies quickly.

G.3.6 Scaling and Parallel Efficiency

Finally, the parallel efficiency of the training was characterized with strong and weak scaling experiments. Both experiments used a global batch size (GBS) of 16.8 M tokens (sequence length 4096) at target scale of 4096 GPUs, and training runs ranged from 8 nodes (32 GPUs), the smallest resource with sufficient memory for the strong scaling experiment, up to 1024 nodes (4096 GPUs). In the weak scaling run the GBS ranged from 0.13 M to 16.8 M tokens (32 to 4096 sequences, *i.e.* proportional to the number of GPUs used), while it was constant in the strong scaling run. The result of this is shown in Figure 15, with ultimately 80% strong scaling parallel efficiency at 4096 GPUs. The performance at this scale is 723 tokens per second per GPU.

G.4 FLOPs Estimation

To estimate the FLOPs used for pretraining, we use a short Python script that accounts for all major operations in the Transformer architecture, provided in Figure 16. Plugging in the 70B configuration (Table 5) at a sequence length of 4096, this results in an estimate of $6.74 \cdot 10^{24}$ FLOPs.

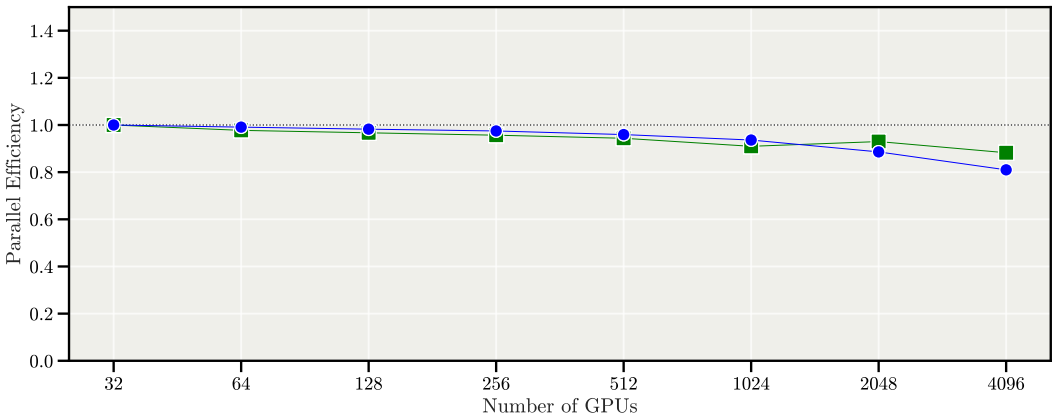

Figure 15: **Scaling of the Apertus 70B model**. Strong scaling parallel efficiency, with the global batch size held constant at 16.8 M tokens, is shown with blue circles. Weak scaling parallel efficiency is shown with green squares, where the global batch size varies from 0.13 M to 16.8 M tokens with increasing GPU count.

H Acknowledgements

This work was supported as part of the Swiss AI Initiative by a grant from the Swiss National Supercomputing Centre (CSCS) under project ID a06 on Alps. We are also grateful to all the many people who have supported and enabled this project, including the management teams of EPFL, ETH Zurich and CSCS, the Hugging Face research team, the PublicAI team, Swisscom, as well as Abdessalam Derouich, Alex Dremov, Anaëlle Touré, Atli Kosson, Chris Wendler, Christiane Sibille, Dan Alistarh, Daniel Dobos, David Atienza, Deniz Bayazit, Fabio Rinaldi, Florian Meyer, Gael Hurlimann, Guilherme Penedo, Helga Rietz, Hynek Kydlíček, Ignacio Pérez Prat and all of Lia Rumantscha, James Henderson, Khadidja Malleck, Leandro Von Werra, Lonneke van der Plas, Loubna Ben Allal, Marcel Salathé, Maria Grazia Giuffreda, Mark Cieliebak, Mary-Anne Hartley, Mateo Muller, Melissa Anchisi, Mrinmaya Sachan, Pascal Frossard, Rico Sennrich, Robert West, Rüdiger Urbanke, Simon Scandella, Stefan Bechtold, Stella Biderman, Timo Kehrer.

```python
def attention_gqa_flops(
    seq_len: int,
    d_model: int,
    key_size: int,
    num_heads: int,
    num_kv_heads: int,
) -> int:
    assert num_heads % num_kv_heads == 0
    heads_per_kv = num_heads // num_kv_heads

    q_proj = 2 * seq_len * d_model * (num_heads * key_size)
    k_proj = 2 * seq_len * d_model * (num_kv_heads * key_size)
    v_proj = k_proj
    qk = 2 * num_heads * seq_len * seq_len * key_size
    qk_norm = qk_norm_flops(seq_len, key_size, num_heads,
            num_kv_heads)
    softmax = 3 * num_heads * seq_len * seq_len
    attn_v = 2 * num_heads * seq_len * seq_len * key_size
    out_proj = 2 * seq_len * (num_heads * key_size) * d_model

    return (
        q_proj
        + k_proj
        + v_proj
        + qk
        + qk_norm
        + softmax
        + attn_v
        + out_proj
    )

def dense_mlp(seq_len, d_model, ffw_size, swiglu=False):
    if not swiglu:
        return 2 * seq_len * (2 * d_model * ffw_size)
    else:
        return 2 * seq_len * (3 * d_model * ffw_size)

def qk_norm_flops(
    seq_len: int, key_size: int, num_heads: int, num_kv_heads: int
) -> int:
    vectors = seq_len * (num_heads + num_kv_heads)
    return 4 * vectors * key_size

def rmsnorm(seq_len, d_model):
    return 4 * seq_len * d_model

def final_logits(seq_len, d_model, vocab_size):
    return 2 * seq_len * d_model * vocab_size

def get_flops(
    n_layers,
    seq_len,
    vocab_size,
    d_model,
    key_size,
    num_heads,
    num_kv_heads,
    ffw_size,
    swiglu=False,
):
    return (
        n_layers
        * (
            attention_gqa_flops(seq_len, d_model, key_size, num_heads,
                    num_kv_heads)
            + dense_mlp(seq_len, d_model, ffw_size, swiglu=swiglu)
            + 2 * rmsnorm(seq_len, d_model)
        )
        + final_logits(seq_len, d_model, vocab_size)
    )
```

Figure 16: **FLOPs computation.** Instead of the common approximation of 6=ND, we use more detailed calculations for the FLOPs estimation based on the Transformer model configuration. We provide the Python code above.